PROGRAMS FOR EVAUATION OF 3D PET RECONSTRUCTION ALGORITHMS

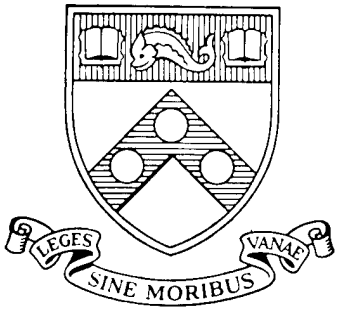
S.S. Furuie[1], S. Matej, G T. Herman, T.K. Nayaran, R.M. Lewitt, P. Kinahan[2] and J.S. Karp[2]

Medical Image Processing Group
[2]Physics & Instrumentation Group
University of Pennsylvania
Philadelphia, PA

[1]Division of Informatics
Heart Institute
Sao Paulo University, Brazil

TECHNICAL REPORT NO. MIPG206

February, 1994



Department of

RADIOLOGY

University of Pennsylvania

PLEASE CHECK THE APPROPRIATE BLOCK BELOW:

-AQ# AQ8-02-0237

☒ 1 copies are being forwarded. Indicate whether Statement A, B, C, D, E, F, or X applies.

☒ DISTRIBUTION STATEMENT A: per Dr Herman
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

☐ DISTRIBUTION STATEMENT B:
DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES ONLY; (Indicate Reason and Date). OTHER REQUESTS FOR THIS DOCUMENT SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT C:
DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES AND THEIR CONTRACTORS; (Indicate Reason and Date). OTHER REQUESTS FOR THIS DOCUMENT SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT D:
DISTRIBUTION AUTHORIZED TO DoD AND U.S. DoD CONTRACTORS ONLY; (Indicate Reason and Date). OTHER REQUESTS SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT E:
DISTRIBUTION AUTHORIZED TO DoD COMPONENTS ONLY; (Indicate Reason and Date). OTHER REQUESTS SHALL BE REFERRED TO (Indicate Controlling DoD Office).

☐ DISTRIBUTION STATEMENT F:
FURTHER DISSEMINATION ONLY AS DIRECTED BY (Indicate Controlling DoD Office and Date) or HIGHER DoD AUTHORITY.

☐ DISTRIBUTION STATEMENT X:
DISTRIBUTION AUTHORIZED TO U.S. GOVERNMENT AGENCIES AND PRIVATE INDIVIDUALS OR ENTERPRISES ELIGIBLE TO OBTAIN EXPORT-CONTROLLED TECHNICAL DATA IN ACCORDANCE WITH DoD DIRECTIVE 5230.25, WITHHOLDING OF UNCLASSIFIED TECHNICAL DATA FROM PUBLIC DISCLOSURE, 6 Nov 1984 (Indicate date of determination). CONTROLLING DoD OFFICE IS (Indicate Controlling DoD Office).

☐ This document was previously forwarded to DTIC on _____ (date) and the AD number is _____.

☐ In accordance with provisions of DoD instructions, the document requested is not supplied because:

☐ It will be published at a later date. (Enter approximate date, if known).

☐ Other. (Give Reason)

DoD Directive 5230.24, "Distribution Statements on Technical Documents," 18 Mar 87, contains seven distribution statements, as described briefly above. Technical Documents must be assigned distribution statements.

Pat Mawby for
G. T. Herman
Authorized Signature/Date

G. T. Herman
Print or Type Name

(215) 898-9841
Telephone Number

19971105 033

PROGRAMS FOR EVAUATION OF 3D PET RECONSTRUCTION ALGORITHMS

S.S. Furuie[1], S. Matej, G T. Herman, T.K. Nayaran, R.M. Lewitt, P. Kinahan[2] and J.S. Karp[2]

Medical Image Processing Group
[2]Physics & Instrumentation Group
University of Pennsylvania
Philadelphia, PA

[1]Division of Informatics
Heart Institute
Sao Paulo University, Brazil

TECHNICAL REPORT NO. MIPG206

February, 1994

Programs for evaluation of 3D PET reconstruction algorithms

S. S. Furuie[1], S. Matej, G. T. Herman, T. K. Narayan, R. M. Lewitt, P. Kinahan[2], J. S. Karp[2]

Medical Image Processing Group
[2]Physics & Instrumentation Group
Department of Radiology, University of Pennsylvania
Blockley Hall, Fourth Floor, 418 Service Drive, Philadelphia, PA 19104–6021,USA
[1]Division of Informatics
Heart Institute, Sao Paulo University, Brazil

Address for correspondence: MIPG/Department of Radiology
Blockley Hall, Fourth Floor
418 Service Drive
Philadelphia, PA 19104–6021

This work was supported by: FAPESP (SP, Brazil) grant 92/0310–1, NIH grants HL28438 and CA54356, and DOE grant DE–FG02–88ER6064.

MEDICAL IMAGE PROCESSING GROUP

TECHNICAL REPORT NO. MIPG206

FEBRUARY 1994

Contents

Chapter 1. INTRODUCTION

Evaluation of a reconstruction algorithm should be done using a sample set that is large enough to provide us with a statistically significant result (Herman and Yeung, 1989). In order to carry out an evaluation, one possibility is to use a set of computer simulated phantoms, that takes into account some parameter variabilities.

This technical report describes in detail programs that generate a set of 3D phantoms and projection data, reconstruct, evaluate and then compare. The main characteristics are:

1. Phantom and projection data generation:
   a. Phantoms with many (69) ellipsoid features, ranging from small (4 mm) to large (40 mm) sized features
   b. Phantoms are random samples from a statistically described ensemble of 3D images resembling those to which PET would be applied in a medical situation (features with random size, orientation and activity)
   c. Features are inside spheres that provide background value for some important clinical tasks such as detectability
   d. Types of features: hot, cold and normal spots
   e. Emulation of 3D PET scanner for projection data generation, with detector field-of-view (FOV) blurring and a realistic 3D PET noise model
2. Reconstruction algorithms:
   a. Algebraic Reconstruction Technique using blob (ARTblob) as basis function (Matej and Lewitt, 1992)
   b. ART using voxels (ARTvox)
   c. EM-ML using blobs (EMblob)
   d. EM-ML using voxels (EMvox)
3. Evaluation for following tasks:
   a. Training figure-of-merit (FOM)
   b. Structural accuracy
   c. Hot spot detectability
   d. Cold spot detectability
4. Statistical comparison using paired t-test

Justifications for using some models (noise, blurring, contrast, ...) are described in the paper (Furuie et al, 1994) and an application evaluating some reconstruction methods is reported in (Matej et al, 1994).

This technical report also describes the usage and assumptions of program "genPhantom3d", that is in public domain. All sources are provided, basically to allow users to compile and run in different computer environments.

These programs were developed using C language (K&R), under a UNIX operating system. They have been tested on SUN SPARCstations (SunOS Release 4.1.3) and Silicon Graphics machines (IRIX Release 4.0.5 System V), however we disclaim any responsibility for possible errors.

Chapter 2. HOW TO INSTALL AND RUN

2.1. Requirements

Disk space needed for sources and executable programs (static) is around 3 MB.

For the default configuration, the program "genPhantom3d" generates at least 3 files:

1. a Phantom Description file with extension **.pde** (5,260 bytes)
2. a phantom file with extension **.img** (2,131,968 bytes)
3. a projection data file with extension **.scn** (33,939,456 bytes)

However a file that contains attenuation factor for all projection rays (phaLengths.AttenF.scn) is also required in order to implement the noise model. It can be generated once simultaneously with the first set of phantoms (see running section). Thus the minimum amount of disk space is around 70 MB.

2.2. Loading the installation files

The tape (1/4 inch, 150MB) contains two tarfiles: installation files and 3 samples of data set.

Procedure:

1. Create a directory (e.g. EVAL3DPET) to be the "evaluation" directory which will be accessible to users.
2. Change directory to 'EVAL3DPET' and load installation files from tape (tarfile), using, for example:
   tar xvf /dev/rst0 .

This procedure will create and load the following directories:

a) sources

b) sources/gen

   gen.mak
   genPhantom3d.c
   MIPGgenPhan.c
   MIPGgenProjs.c
   MIPGphaFunc.c

c) sources/rec

   rec.mak
   ARTbloPet.c
   ARTvoxPet.c
   em_blpet.c
   em_vopet.c
   blob_fns.c
   basisfns.c
   besselfns.c
   voxel_fns.c

d) sources/bib

bib.mak
MIPGmath.c
MIPGstring.c
MIPGutil.c
imagio1.1.c

e) sources/eval

eval.mak
eval3d.c
compImgs.c
student.c

f) sources/util

util.mak
createPDE.c
readPDE.c

g) include

genPhantom3d.inc
MIPGdefs.h
MIPGgenPhan.h
MIPGphaFunc.h
MIPGgenProjs.h
MIPGstring.h
MIPGmath.h
MIPGutil.h
blob_fns.h
voxel_fns.h
imagio.h
pet_hdr.h

h) bin

buildall.bat
recon.bat
eval.bat
student.bat

The second tarfile contains 3 sets of phantoms (phaMIPG000.*, phaMIPG001.* and phaMIPG010.*, where *=**.pde**, **.img**, **.scn**) and attenuation factor file (phaLengths.AttenF.scn). It requires around 145 MB. It will create directory 'phantoms'.

2.3. Compiling

Once the installation files have been loaded, there is a script that allows compilation and linking of all programs. This procedure has been tested on the SUN and Silicon Graphics workstations and should work with most Unix systems.

Procedure:

1. Set environment variable 'dir3d' to the "evaluation" directory
   (Eg.: setenv dir3d /usr/EVAL3DPET)
2. Change directory to bin.
3. Run the script buildall.bat, for example with parameter 'sparc':
   buildall.bat sparc

It will create (if not already there) directory **sparc** and create all executables there.

2.4. Running

Line commands:

1. Phantom/projection generation:
   genPhantom3d <prefix> <start#> <numPhantoms> <totalCounts> <FLAGatten>

   a. **prefix**: prefix of filenames to be created. Eg.: phant
   b. **start#**: first number to be appended to prefix. Eg.: 100
   c. **numPhantoms**: number of phantoms to be created. Eg.: 5 (files phant100.* to phant104.* will be created)
   d. **totalCounts**: total detected counts considering all detectors. Eg.: 80000000
   e. **FLAGatten**: if 1, it will also create file with attenuation correction factor (phaLengths.AttenF.scn)

   typical CPU (SUN sparc10) time for one phantom/projection data generation (complete, with blurring and noise): 8 h.

   If you want to create phantom or projection data for different reconstruction volume, phantom geometry or scanner geometry you should modify the file " genPhantom3d.inc" (see Appendix C) prior to compilation. This file contains global definitions for program "genPhantom3d".

2. Reconstruction programs:
   For all following reconstruction programs, it is assumed that the input projection data are without attenuation. Projection file (**.scn**) contains data that were already corrected for attenuation (PET data model), or contains estimates of line integrals (CT data model). (Time information is for 1 iteration, using SPARC 10, see Matej et al, 1994.)

   a. ARTvox:
      Usage: **artvox** [Options and Parameters]
      Options:
      -h for this message

Parameter Values: (note space before value)
-b (Batch mode - program exits if wrong parameters)
-i namepr (name of input projection file)
-f name (name of output/input data & image, extensions are added)
-s startflag (Start from : 1:data on disk; 2:const(avg) cylinder; 3:zero)
-n noit (number of iterations)
-l lambda (relaxation factor (0.0 - 2.0))

reconstruction time (1 iteration): 2h 11min

b. ARTblob:
Usage: **artblob** [Options and Parameters]
Options:
-h for this message
Parameter Values: (note space before value)
-b (Batch mode - program exits if wrong parameters)
-i namepr (name of input projection file)
-f name (name of output/input data & image, extensions are added)
-s startflag (Start from : 1:data on disk; 2:const(avg) cylinder; 3:zero)
-n noit (number of iterations)
-l lambda (relaxation factor (0.0 - 2.0))
-r blrad (blob radius)
-m blord (order of Bessel function in blob)
-a blalpha (blob parameter alpha)
(Typical/Standard blob: -r 2.0 -m 2 -a 10.4)

reconstruction time (1 iteration): 10h17min

c. EMvoxel:
Usage: **emvox** [Options and Parameters]
Options:
-h for this message
Parameter Values: (note space before value)
-b (Batch mode - program exits if wrong parameters)
-i namepr (name of input projection file)
-f name (name of output/input data & image, extensions are added)
-s startflag (Start from : 1:data on disk; 2:constant cylinder)
-n noit (number of iterations)
-t nameattpr

(filename - of attenuation coefficients on proj.lines (program attenuates projections and considers attenuation also in EM-ML algorithm),

0 - if CT or no attenuation of PET data considered.)

[-c corname (base name of corr. matrix file - if other than standard)]
reconstruction time (1 iteration): 3h10min

d. EMblob

Usage: **emblob** [Options and Parameters]

Options:

-h for this message

Parameter Values: (note space before value)

-b (Batch mode - program exits if wrong parameters)

-i namepr (name of input projection file)

-f name (name of output/input data & image, extensions are added)

-s startflag (Start from : 1:data on disk; 2:constant cylinder)

-n noit (number of iterations)

-r blrad (blob radius)

-m blord (order of Bessel function in blob)

-a blalpha (blob parameter alpha)

(Typical/Standard blob: -r 2.0 -m 2 -a 10.4)

-t nameattpr

(filename - of attenuation coefficients on proj.lines (program attenuates projections and considers attenuation also in EM-ML algorithm),

0 - if CT or no attenuation of PET data considered.)

[-c corname (base name of corr. matrix file - if other than standard)]

reconstruction time (1 iteration): 9h 10min

For programs 'artvox', 'artblob', 'emvox', 'emblob', one can see the running status by printing file 'name'.stat. The file 'name'.hist gives information about parameters used for reconstruction. Each of these 4 programs can be used sequentially. For example, in order to use 'emvox' and store results at 10th and 30th iteration, with starting image as a constant cylinder (-s 2), one could use the following script:

```
# Reconstruct first 10 iterations and evaluate
emvox -b -i phantom.scn -f phantomEmvox -s 2 -n 10 -t 0
eval3d phantom.img phantomEmvox.img phantomEmvox.eval 4

# Reconstruct next 20 iterations and evaluate
emvox -b -i phantom.scn -f phantomEmvox -s 1 -n 20 -t 0
eval3d phantom.img phantomEmvox.img phantomEmvox.eval 4
cat phantomEmvox.hist >> phantomEmvox.eval
```

All these 4 programs assume that there is a **.pde** file associated to projection data file. In the example above, it is assumed that 'phantom.**pde**' exists. It is possible to create a '.**pde**' file using program **createPDE** (see item 4).

3. Evaluation:

a. **eval3d** <ideal image filename> <reconstructed image filename> <evaluation result filename> <DETECT> [FLAG]
where:
DETECT: detectability index
0: equation (4) [see section 3.3]
4: equation (5) [see section 3.3]
FLAG (optional):
0: default (prints only numerical values)
1: prints ROC global curve
2: prints ROC global & partial curves
eval3d utilizes **.pde** file of <ideal image filename> to get information for evaluation (dimensionalities, position, size and activity of features,...).
This program appends (creates if not exist) results of evaluation to the file <evaluation result filename>. In order to facilitate further analysis (such as statistical comparison) it also creates a file <evaluation result filename>.**tbl** that contains (cumulatively) a table of calculated figure-of-merit (FOMs). For each call to **eval3d** it stores in the **.tbl** a row with:
<reconstructed image filename> <Training FOM> <SA_gl> <HSD_gl> <CSD_gl> <SA_S> <SA_M> <SA_L> <HSD_S> <HSD_M> <HSD_L> <CSD_S> <CSD_M> <CSD_L> <HSD_avg> <CSD_avg>
where Training FOM:equation 7,
SA: structural accuracy (equation 3),
HSD: hot spot detectability using ROC area,
CSD: cold spot detectability using ROC area.
The suffix means:
_gl : global value, considering features of all sizes,
_avg: averaged ROC area, calculating separately ROC area for small (_S), medium (_M) and large (_L) sizes,
_S: small features (surrounding sphere radius {4.0 , 6.4} mm),
_M: medium size features (surrounding sphere radius {8.0 , 10.0} mm),
_L: large size features) (surrounding sphere radius {16.0 , 20.0} mm).
There are high similarities among some of FOMs (see appendix G) for instance between HSD_gl and HSD_avg using equation 4 or between HSD_avg using equation 4 and 5.

b. **compImgs** <reference.img> <imageToBeCompared.img>

It calculates minimum, maximum, average and total counts for <reference.img>, <imageToBeCompared> and (<reference.img> — <imageToBeCompared>).

c. **student** <fileTbl1> <fileTbl2> <numbRows> <numCols> <column[0..]> [<FLAG>]

It compares (student statistical test) columns (<column>) of two tables (or same table) in files <fileTbl1> and <fileTbl2>, each with <numbRows> rows and <numCols> columns. [FLAG] is optional and if equal to 1, it prints details of calculation. See Appendix F for examples.

4. Others:

a. **createPDE**

It is a simple program to create a **.pde** file. Sometimes we have a projection data (such as Hoffman phantom) in Petview format and we want to reconstruct it using above mentioned iterative methods. In this case we have to create a **.pde** file.

b. **readPDE**

It reads and displays the content of a **.pde** file

c. **recon.bat, eval.bat, student.bat**

Samples of scripts (see Appendix F) for reconstruction, evaluation and statistical comparison in batch.

Chapter 3. FRAMEWORK

In this chapter we describe the phantom model, projection data generation, some figures of merit for evaluation and testing. For a complete description with justifications, see (Furuie et al, 1994).

3.1. Phantom generation

In order to evaluate fully 3–D PET reconstruction algorithms, we are using an ensemble of phantoms, which are roughly related to the task of imaging features of relatively high or low activity (as compared to their background) in the head. The "head" is an ellipsoid (with radii R_x = 128.0 mm, R_y = 96.0 mm, R_z = 147.8 mm, and center at z = 64 mm, see Fig. 1), all of which is assumed to contain attenuating material, but only a part of which (the part which lies in the "reconstruction region" indicated in Fig. 1) is assumed to have activity in it. The activity in that part is assumed to be uniform (and, since we are free to choose our units, we consider this uniform activity to have value 1) except for 69 ellipsoid-shaped features. Although the features themselves change in size and shape from phantom to phantom (as is explained below), each of the features is located inside one of 69 spheres, which themselves are the same in all the phantoms. The geometrical distribution of the spheres is indicated in Fig. 1 and their sizes are shown in Table I. Precise location and size of spheres are given in Appendix D. As indicated on the right side of Fig. 1, the spheres are located in three layers (A, B, and C). Each layer contains 11 larger spheres, whose centers are distributed according to the schematic:

	01	02	03	
04	05	06	07	08
	09	10	11	

In the middle of the segments that connect the center of sphere '04' to that of '01' or of '09' and that connect the center of sphere '08' to that of '03' or of '11', there are smaller spheres (see left side of Fig. 1). Immediately above and below (in the z-direction) each of these smaller spheres there are two other spheres of the same radius and so we have altogether 12 smaller spheres per layer. The centers of the layers C, B, and A are at heights –50 mm, –16 mm, and 34 mm, respectively.

The features are generated using the following random process:

• The shape of a feature is an ellipsoid whose centre is at the center of one of the spheres discussed in the previous paragraph. Each of the three radii of the ellipsoid is independently and randomly selected

Table I. Radii of the spheres containing features, contrast of the features in them if they are "hot" or "cold," their location, and the total number of spheres of each type in a phantom.

Radius	4.0 mm	6.4 mm	8.0 mm	10.0 mm	16.0 mm	20.0 mm
Contrast	75.0%	29.3%	18.8%	12.0%	4.7%	3.0%
Layer	C	B	A	C	B	A
Number	12	12	12	11	11	11

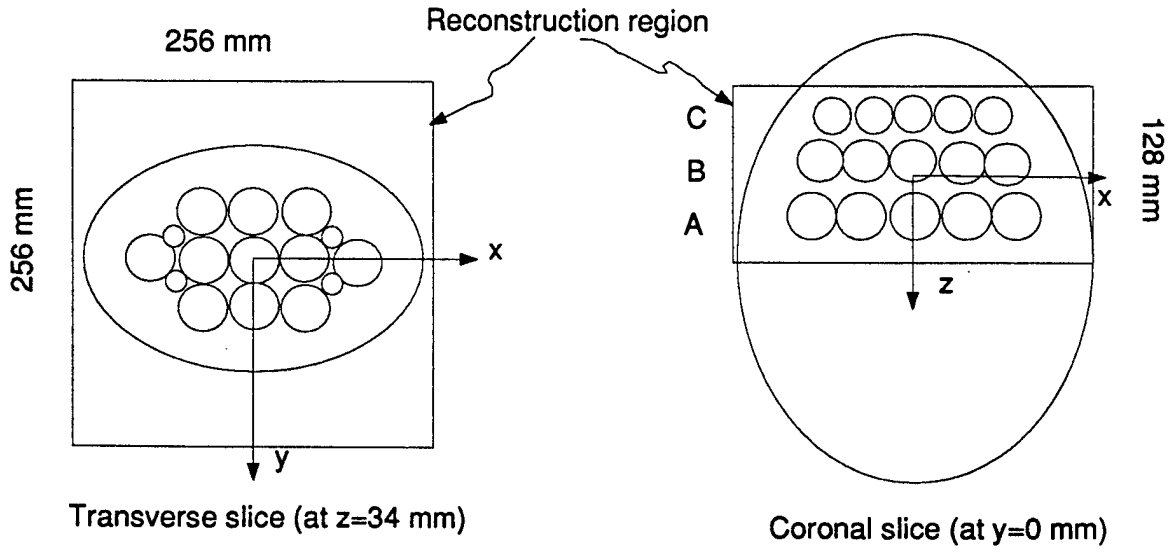


Figure 1. Illustration of the shape of the reconstruction region in our phantoms and of the geometrical distribution of the spheres which contain the randomly-shaped features.

from a distribution which is uniform between 50% to 100% of the radius of the enclosing sphere. Orientations of the ellipsoid axes (rotations around the three coordinate axes) are also independently and randomly selected from a distribution which is uniform between 0 and 180 degrees. The part of the enclosing sphere which is outside the feature is referred to below as the "background" of the feature. (Note that since we have features of varying sizes, our methodology can be adapted to allow us to evaluate reconstructions from the points of view of FOMs specialized to features in a certain size range.)

• Each feature is randomly assigned to be one of three equiprobable types: NORMAL (the activity in the feature is the same as in its background), HOT (the feature is a "hot spot"), or COLD (the feature is a "cold spot"). The activity assigned to hot spots (respectively, to cold spots) is higher (respectively, lower) than the activity in their background by the contrast listed in Table I. This contrast varies from 3% to 75% (of the background activity), depending on the size of the enclosing sphere. Such variation in contrast is necessary in order to have approximately the same difficulty in detecting features with different sizes (see Appendix A). The actual size of the contrast (only 3% for large features and 75% for very small features) is much less than what one would like to have for reliable detection of features. However, this is essential for comparing reconstruction algorithms; if all features had high contrast, then we would have nearly 100% detectability associated with all reasonable methods and we would not be able to determine their relative merit for hot spot and cold spot detection. (We do not include a phantom in our ensemble unless it has at least one feature of each type.) For the evaluation computation, we consider that a voxel is "in a feature," if its center and at least five of its vertices are in the feature (i.e., they lie in the ellipsoid which is the support of the feature). We consider that a voxel is "in the background of a feature," if its center and at least five of its vertices are in the sphere enclosing the feature but not in the feature. If the random generation of a phantom resulted in it containing a feature such that either the feature itself or its background contained fewer than five voxels, then we generate another feature and check again.

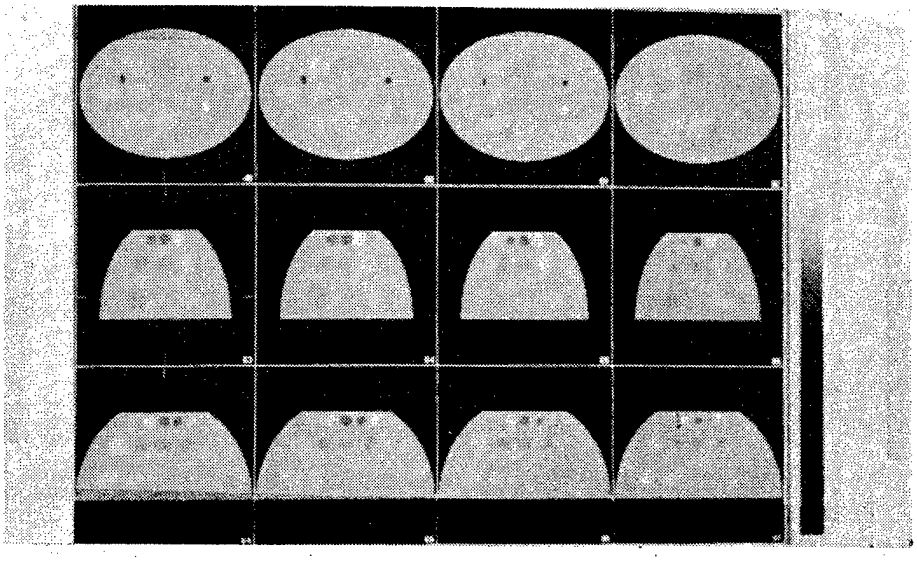


Figure 2. Slices through one of the randomly generated phantoms (first row: transverse; second row: sagittal; third row: coronal).

For comparisons with reconstructions, each phantom is digitized as an array with $64\times128\times128$ cube-shaped voxels, each with 2mm side length. The value assigned to a voxel in a phantom is determined as the average of the values of activity at the regularly-spaced points of a $4\times4\times4$ array within the voxel. Slices through the digitized version of one of our randomly generated phantoms are shown in Fig. 2.

3.2. Projection data generation

For the generation of projection data, we simulate an actual 3–D PET scanner with a cylindrical detector of 420 mm in diameter and 256 mm in axial length; see Fig. 3. (Such a scanner is currently under construction in our department.) We are assuming that data are rebinned into sets of parallel raysums, so that, for each "view," we have a virtual panel of 90×128 measurements corresponding to "bins" perpendicular to the panel. This panel "rotates" 180 degrees around the reconstruction volume producing a total of 96 views per tilt position. We generate 15 steps of panel tilting between -26.5 to 26.5 degrees (step = 3.75 degrees). It is assumed that the center line (axis for tilting) of the panel is in the central plane of the reconstruction volume (z=0) and contains the origin of coordinate system. Thus the total number of bins is $15\times96\times90\times128$ (16.6 million). The cylindrical nature of our simulated detector implies that raysums are not available for some of the bins in the tilted views. In calculating the projection data for those bins for which such data are available, the cross-section (2×2 mm) of the bin is sampled by a square arrangement of four rays (parallel to and 1 mm apart from each other). For each of these rays the integral of activity is calculated analytically. These calculated integrals are

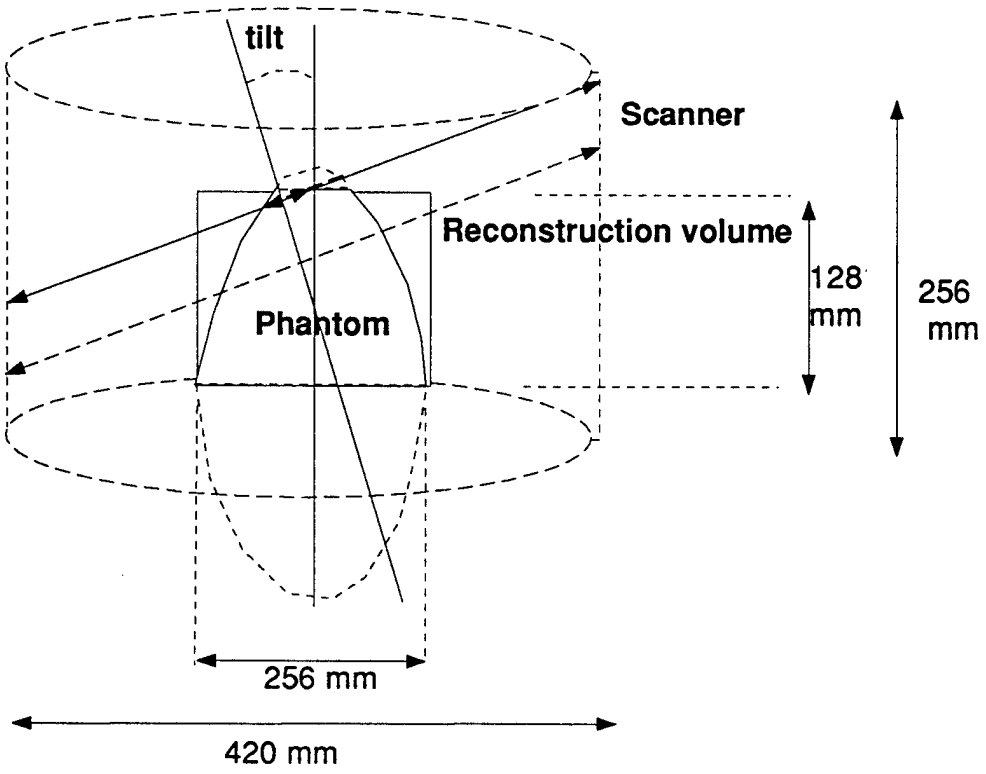


Figure 3. Illustration of scanner geometry. The reconstruction volume is centered in the scanner.

blurred by a Gaussian smoothing filter with a full-width at half-maximum (FWHM) approximately equal to 4 mm, to match the combined effects of the uncertainty in the exact location of a detected event and of the rebinning from actual measurements by such a scanner into the arrangement of panels that we use in the reconstructions. The blurred integrals which are associated with the four rays of a bin are averaged to provide us with the "raysum" of activity associated with that bin.

Because of attenuation, the measurement (detected counts) associated with a bin will be substantially less than its raysum. We take the expected value of the detected counts for a particular bin to be p/k, where p is the raysum associated with the bin and the attenuation factor k is calculated to be $exp(L\mu)$, where L is the analytically calculated length of intersection (in mm) of the central ray of the bin in question with the phantom and μ is the linear attenuation coefficient of water (0.0095 mm^{-1} for 511 keV). For our simulations, we assumed the total number of detected counts to be 80 million, which represents nearly 56,000 per projection plane and averages to be around 5 per bin. The actual range of the expected number of detected counts per bin for our phantoms turns out to be from 0 to 27. After attenuation correction, the total number of counts in sinogram (all panels) is around 335 million.

Our calculation of the noise in the projection data takes into consideration that corrections for attenuation, scattering, detector normalization, etc. cause a discrepancy between the noise in actual data and the noise that would be provided by a pure Poisson model (see, e.g., Rowe and Dai 1992). For any calculated raysum p, we replace it by its noisy version, which is calculated as

$$k \times \left[\text{Poisson}\left(\frac{p}{k}\right) + N(0, s^2)\right], \tag{1}$$

where Poisson(p/k) denotes a sample from a Poisson distributed process with mean p/k (the attenuated raysum) and $N(0, s^2)$ denotes a sample from a zero-mean normal distribution with standard deviation

equal to s (whose actual value, as we show below, depends on p/k in a well-defined way). This second random distribution forms part of our noise model to account for discrepancies between actual physical measurements and our model of them. The methodology for choosing s in conjunction with the noise model implied by the form of equation (1) was determined experimentally using cylinder-shaped physical phantoms (Furuie et al, 1994):

$$s^2 = \frac{p}{k} \times \left[\left(1 + 0.05 \sqrt{\frac{p}{k}} \right)^2 - 1 \right] . \tag{2}$$

Our projection data generation process can be summarized as follows: raysums (p) of activities are first generated using analytical integrations of the geometrically described phantom and a blurring process and this is then followed by the introduction of noise according to equations (1) and (2). Samples from one of the resulting projection data sets are shown in Fig. 4 for tilts at 0 and at –26.5 degrees. For the latter case, we also see the effects of not having measurements for a large number of bins due to the cylindrical nature of our detector.

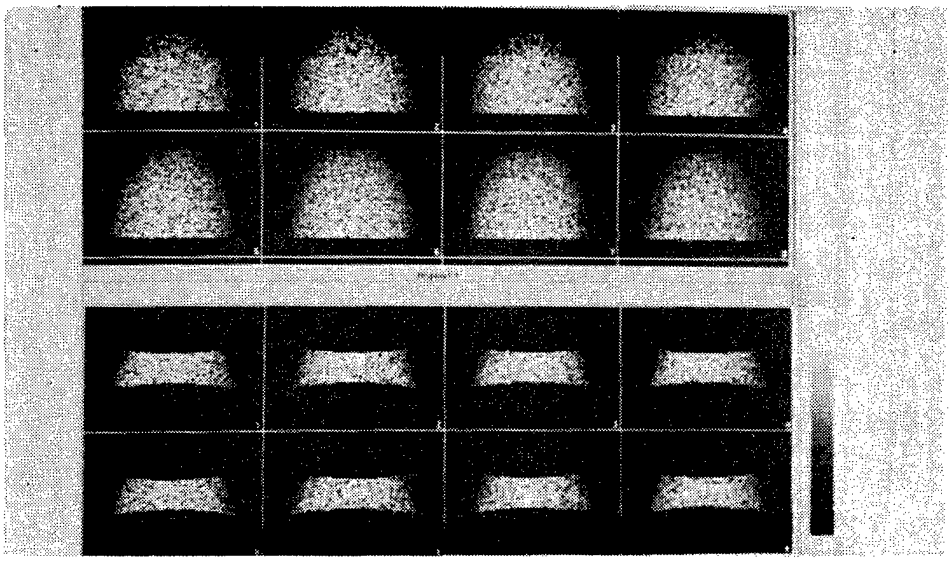


Figure 4. Samples of projection data for views with tilt of 0 degrees (top) and with tilt of —26.5 degrees (bottom). Observe that data are not available for quite a few bins in the latter case

3.3. Figures of merit

The definition of a figure of merit (FOM) should be related to the task of solving a medical problem. Prior to giving some examples, we discuss some notation common to all of them. In this notation we assume a single phantom and a single reconstruction (by some method) from a projection

data set of that phantom. An FOM is a measure of how good that particular reconstruction is from some mathematically precise point of view.

Consider a phantom which contains F features. (In all our phantoms F is 69, but the definitions of the FOMs are applicable to a much larger class of phantoms.) We use f to refer to the fth of the F features. For this particular feature, we use R_f to denote the radius of the surrounding sphere, μ_f to denote the average activity of the voxels within the feature in the digitized phantom, n_f to denote the number of voxels in the feature, m_f and v_f to denote the average and the variance of the activity within the feature in the reconstruction, $\underline{n_f}$ to denote the number of voxels in the background of the feature, and $\underline{m_f}$ and $\underline{v_f}$ to denote the average and the variance of the activity within the background of the feature in the reconstruction.

We define the FOM *structural accuracy* of a reconstruction as the value of

$$1 - \frac{1}{F}\sum_{f=1}^{F}\left(\frac{R_f}{R_0}\right)^{\frac{3}{2}}\left|m_f - \mu_f\right|, \tag{3}$$

where we think of the normalizing parameter R_0 as the radius of a reference sphere. Specifically, for our phantoms we select R_0 to be 9.0 mm, which is an intermediate size of our range of radii (see Table I). Note that more accurate average activity values in the features in the reconstruction will result in a higher value of structural accuracy, with a perfect reconstruction yielding the value 1 for this FOM. (For our phantoms we also found, see Matej *et al* 1994, that practical reconstruction algorithms yield structural accuracy values in excess of 0.5 and so in the same range as provided by other FOMs that we discuss below.) The sum in (3) is a "weighted 1-norm" for the average activities within features, where the weights account for the different sizes of the features. These weights were derived from the following considerations. If we assume that the variance of the noise in a single voxel of the reconstruction is the same for all voxels, then the variance of the noise in m_f is inversely proportional to n_f and so, on average, to R_f^3. Therefore, under the circumstance that in each individual feature the standard deviation of noise is the same for all voxels in the feature, our chosen weight factors would yield a weighted 1-norm independent of size, which is proportional to the average of the standard deviations of error in the individual features. The value of the unweighted 1-norm would be influenced mainly by the noise in the smaller features; by raising the normalizing factor to the power 3/2 we equalize the expected influence of features of various sizes.

For the next two FOMs (Hot Spot Detectability and Cold Spot Detectability) we need an intermediate concept. With every feature f in a reconstruction we associate a detectability index h_f. One possibility is to use index

$$h_f = \frac{\sqrt{n_f.\underline{n_f}}\left(m_f - \underline{m_f}\right)}{\sqrt{n_f.v_f + \underline{n_f}.\underline{v_f}}} \tag{4}$$

which is based on one of the standard measures of separability (between the reconstructed voxel values in the feature and the reconstructed voxel values in the background of the feature) in statistical pattern recognition theory (Fukunaga 1972)[see Appendix B for details].

Note that in a good reconstruction, we would expect this index to be positive if the feature is a hot spot, negative if the feature is a cold spot, and approximately zero otherwise. The absolute value of the index goes up if the absolute difference in the means of voxel values in the feature and in its background gets greater, but goes down if the variance in these voxel values goes up.

In the evaluation of the practical reconstructions there were several problems with the detectability index (4) derived on the basis of idealistic theoretical considerations. First, in order to obtain comparable detectability for the features of different sizes, the feature contrast has to be changed more than suggested by the formula (23), which was considered in derivation of index (4). This is caused by reconstruction problems, like border effects, which are dependent on the feature size. Secondly, since noise in the reconstruction is correlated on the local neighborhood, we are not able to accurately estimate the local variances (v_f, $\underline{v_f}$) for small features. Another possibility is to approximate them by global variance for given reconstructed image. In this case, the variances (v_f, $\underline{v_f}$) will have the same value for all features in the given reconstruction. Therefore, their value will not influence the ROC analysis result and they can be omitted from the detectability index. Thirdly, in the practical reconstructions using several reconstruction algorithms, it has been found that the size dependent correction factor, in detectability index, given by the contrast change does not have desirable effect. Practically, it has been found that the best results (the closest approximation to the independence of expected value and variance of h_f on the size of features) were obtained using the h_f without any correction at all. Considering second and third points above, the best results were obtained in our practical evaluations using following detectability index:

$$h_f = \left(m_f - \underline{m_f}\right) \tag{5}$$

This detectability index turned out to be less sensitive to feature size for the given ensemble of phantoms. Therefore, equation 5 is preferable for ROC analysis using our phantoms. The evaluation program **eval3d** provides ROC area considering both detectability indices and also ROC area per size (see section 2.4.3).

We define the FOM *hot spot detectability* of a reconstruction as the area under the ROC (Receiver Operating Characteristic) curve based on the index h_f (Swets and Pickett 1982, Green and Swets 1988, Herman and Yeung 1989). To be exact, we do the following. Suppose that there are M hot spots and N normal features in the phantom under consideration. For any real number h, let $c(h)$ denote the number of HOT features f for which $h_f \geq h$. Let $f_1, ..., f_N$ be the NORMAL features, with the notation assigned so that $h_{f_1} \geq h_{f_2}, ..., h_{f_{N-1}} \geq h_{f_N}$. Then hot spot detectability is defined to be

$$\frac{1}{MN} \sum_{n=1}^{N} c(h_{f_n}) \, .$$

It is easy to see that this FOM has a value between 0 and 1 and that the value gets near the high end if the indices h_f for normals tend to be small as compared to those for hot spots. In fact, for any reasonable reconstruction method, we expect the hot spot detectability to be greater than 0.5. (A method which randomly assigns values to voxels should end up with hot spot detectability approximately 0.5.)

Except for obvious minor changes, *cold spot detectability* of a reconstruction is defined along the same lines.

The three FOMs defined so far have been designed to reflect various medical tasks: structural accuracy is a measure of how well we can estimate the total uptake in features and the other two indicate how well we can detect hot spots and cold spots, respectively. We complete our list of FOMs with a measure which is less relevant from the medical point of view, but which we have found useful for training algorithms (see the next section and Matej *et al* 1994). For this reason we call it the

training FOM. It is defined as

$$1 - \frac{1}{F}\sum_{f=1}^{F} e_f \,, \tag{7}$$

where e_f is the average over voxels in the fth feature of the absolute difference between voxel values in the reconstruction and in the phantom.

3.4. Training and testing

Many reconstruction algorithms have some free parameters in them (such as relaxation parameters and the number of iterations in iterative methods or filter characteristics in transform methods, Herman 1980). To insure that a general reconstruction method which has some free parameters is not unjustly discarded as a result of a bad choice of these free parameters in an evaluation study, it is appropriate that before testing the method one finds those values of the free parameters for which it performs optimally (or at least well) according to some relevant FOM. This is referred to as *training*. In our environment training can be done as follows.

We randomly generate from our ensemble a few phantoms and their associated projection data; we refer to this as the *training set*. We then search for values of the free parameters which optimize the training FOM for the training set. (In practice, we limit our search until we find values of the free parameters which are such that analysis of changes in the training FOM with changing free parameters indicate that we are within a few percent of the optimal value.) The free parameters then are fixed at these values for use in the testing process described below. It is reasonable in this numerical observer approach to use a different FOM for training and for testing. This is because the performance of algorithms can sometimes be fine-tuned to a very high level by the use of training on the same FOM as is used for testing (Herman and Yeung 1989). This is likely to be unrealistic for a clinical application, where we would not have the luxury of being able to use a different most-appropriately optimized reconstruction algorithm for each of the different tasks.

The evaluation should be performed using a *testing set* of phantoms and projection data which is from the same ensemble as, but is statistically independent of, the training set. Each reconstruction algorithm that is being compared should be applied to all the projection data and FOMs should be calculated for each of the reconstructions.

Once an FOM is calculated for each reconstruction produced by two different reconstruction algorithms, we assign a level of statistical significance to rejecting the null-hypothesis that the two reconstruction methods are equally good (from the point of view of the FOM), in favor of the hypothesis that the one with the higher FOM is better, as follows. Let β_b and δ_b be the FOMs of the reconstructions from the bth of a total of B data sets, as reconstructed by the two methods, respectively. Then, according to the null-hypothesis, $\beta_b - \delta_b$ is a sample of a zero-mean random variable. It follows, for large enough B, that

$$\sum_{b=1}^{B} (\beta_b - \delta_b) \tag{8}$$

is a sample of a normally distributed zero-mean random variable (Mould 1989). The variance of this random variable is B times that of the zero-mean random variable of which $\beta_b - \delta_b$ are samples for

$1 \leq b \leq B$. Hence, for large enough B, it is reasonable to assume that the null-hypothesis implies that (8) is a sample from a normally distributed random variable with mean zero and variance

$$\sum_{b=1}^{B} (\beta_b - \delta_b)^2 .$$

We can thus use the normal distribution to calculate significance (Mould 1989).

An alternative way of assigning significance, which may be preferred when B is considered not large, is by using the t-test for paired data with the value of t calculated according to the following formulas (Mould 1989):

$$t = \frac{m}{s/\sqrt{B}} , \text{ where}$$

$$m = \frac{\sum_{b=1}^{B} (\beta_b - \delta_b)}{B} , \text{ and} \tag{10}$$

$$s^2 = \frac{1}{B-1} \sum_{b=1}^{B} ((\beta_b - \delta_b) - m)^2 .$$

The calculated t (eq. 10) was compared to Student table for DF (degree of freedom) equal to *(B-1)*, for a statistical level of significance (5%, 0.5% and 0.05%).

3.5. Implemented Reconstruction Algorithms

The iterative reconstruction algorithms may be perceived as attempts to solve, in some sense, the system of equations

$$y = Ax,$$

where y is the vector of projection data, A is a matrix whose entries are the contributions of single image basis elements to the data associated with particular bins, and x is the vector of the unknown image values (coefficients in the decomposition of the image into the basis elements). Traditionally, the discrete model of an iterative reconstruction method assumes voxel basis functions (i.e., basis function whose value is 1 inside a specific cube-shaped voxel and is 0 everywhere else). An alternative approach is discussed in (Lewitt 1990, Lewitt 1992), in which images are represented as linear combinations of spherically symmetric basis functions. We have some preliminary experience with the behavior of the EM-ML algorithm using smooth (continuous value and derivative at the basis function "border") spherically symmetric functions (*blobs*, Matej and Lewitt 1992). These experiments have shown that incorporation of blobs into the reconstruction model can be advantageous, especially for the case of projection data with limited resolution (smoothed by physical processes during measurement). Such smoothed projection data have better consistency with the reconstruction model incorporating blobs. Moreover, this type of image representation can be advantageous also from the point of view of 3–D visualization algorithms (Udupa and Herman 1991). We implemented both voxel-based and blob-based versions of EM-ML and of ART for the fully 3–D case.

Specifically, the blobs used in the reconstructions in this paper are based on the generalized Kaiser-Bessel window functions having the form (Lewitt 1990)

$$b_{m,a,\alpha}(r) = \begin{cases} \dfrac{\left[\sqrt{1-(r/a)^2}\right]^m I_m\left[\alpha\sqrt{1-(r/a)^2}\right]}{I_m(\alpha)}, & 0 \le r \le a \\ 0, & \text{otherwise} \end{cases},$$

where r is the radial distance from the blob's center, I_m denotes the modified Bessel function of the order m, a is the radial dimension of the blob and α is the "taper" parameter controlling the blob shape. The following blob parameters (Matej and Lewitt 1992) were used in our experiments: m = 2 (continuous derivative at the blob borders), radius a = 4.0 mm (which is twice the length of a voxel edge), and taper parameter α = 10.4 (chosen so that the radial frequency spectrum has value zero at sampling frequency and is also bounded by a function that decays particularly fast for higher frequencies). At the time of originally designing the experiments, we decided (without considering alternatives) to place the blobs in the three-dimensional space so that the blob-centers are at the same locations as the voxel-centers.

The EM-ML method starts with an image vector x^0, all of whose components have the same positive value, and then proceeds with iterative steps given by the formula:

$$x_j^{k+1} = \frac{x_j^k}{\sum_{i=1}^I a_{i,j}} \sum_{i=1}^I \frac{y_i}{< a^i, x^k >} a_{i,j},$$

where x_j^k is the jth element of the kth iterative estimate of the image vector, y_i is the data item associated with the ith projection bin, $a_{i,j}$ is the contribution of the jth basis function to this data item, and $< a^i, x^k >$ is the inner product (forward projection) of the ith row of the matrix A and the kth iterative estimate of the image vector. In our implementation of the EM-ML method, the system of equations was set up so that y_i is the actual number of detected counts associated with the ith projection bin. This means that in determining entries of A we have to take into consideration the attenuation that takes place in the object whose activity distribution we wish to reconstruct. This is the appropriate model, since the EM-ML method as described above converges to the maximum likelihood estimator of the image vector only if the y_i are samples from Poisson distributions whose means are $< a^i, x >$. (By the same token, the system should contain equations for only those bins for which there is a collected measurement.) In practice we have to stop the algorithm at some point; the parameter that we leave free in our initial description of the EM-ML method is the number of iterations at which the process is terminated.

The ART method starts with an image vector x^0, all of whose components have the same positive value (which is the mean value over the reconstruction region as estimated from the projection data), and then proceeds with iterative steps given by the formula:

$$x_j^{k+1} = x_j^k + \lambda \frac{y_i - < a^i, x^k >}{\sum_{j=1}^J a_{i,j}^2} a_{i,j},$$

where λ is the so-called relaxation parameter, which is the parameter we leave free at this initial stage of describing this algorithm, and everything else is interpreted as it was done for EM-ML. (In ART, just as in FBP, y_i is the estimated integrated activity associated with the ith bin; this is

obtained from the detected counts by a correction for attenuation, Furuie *et al* 1994.) Dissimilarly to the EM-ML method (in which the estimated image vector is updated simultaneously based on all the measurements), each update in ART of the estimated image vector is based on a single projection bin, and the method cycles through all the bins one after another. This leads to a faster initial approach to an acceptable solution and consequently to a lower number of necessary total cycles through the projection data. (In the experiments described in (Matej et al., 1994), only one cycle through the projection data was used, except that it started with and repeated at the end corrections for the bins associated with untilted planes. For the particular mode of data collection that was simulated, this means that the reconstruction process cycles 1.07 times through all of the projection data.) During cycling through the data we use a special ordering of the bins (Herman and Meyer 1993), which results in a rapid early progress towards an acceptable solution. Corrections are made for only those bins for which there is a collected measurement. We fixed the way we cycle through the data for fairness of comparison between the various basic approaches: we wished to have only one free parameter for which we optimize during training. Our experience with ART indicated to us that more is likely to be gained in fixing the number of iterations and optimizing for the relaxation parameter than doing the opposite. We expected that a very small number of cycles through the data will give acceptable results (Herman and Meyer 1993).

Chapter 4. IMPLEMENTATION DETAILS

In this chapter we describe in details the phantom model, reconstruction volume within which we do our reconstruction, the digitization of this volume into a voxel space, the arrangement of lines for which projection data are to be generated, and the method of raysum generation, including noise.

4.1. Reconstruction volume

1. It is assumed that the reconstruction volume (fig. 5) is a cylinder with:
   Dxy=256 mm (diameter),
   Hz=128 mm (height).
2. The origin of a Cartesian coordinate system (xyz, right hand convention) is located at the center of the cylinder, where the z-axis coincides with the cylinder axis (figure 5). In order to be compatible with medical notation for images (transversal, sagittal and coronal views), the direction of z-axis is from top to bottom. More specifically, the patient is laying on xz-plane, with feet in +z direction, looking at +y-axis direction, and with left ear in +x direction.

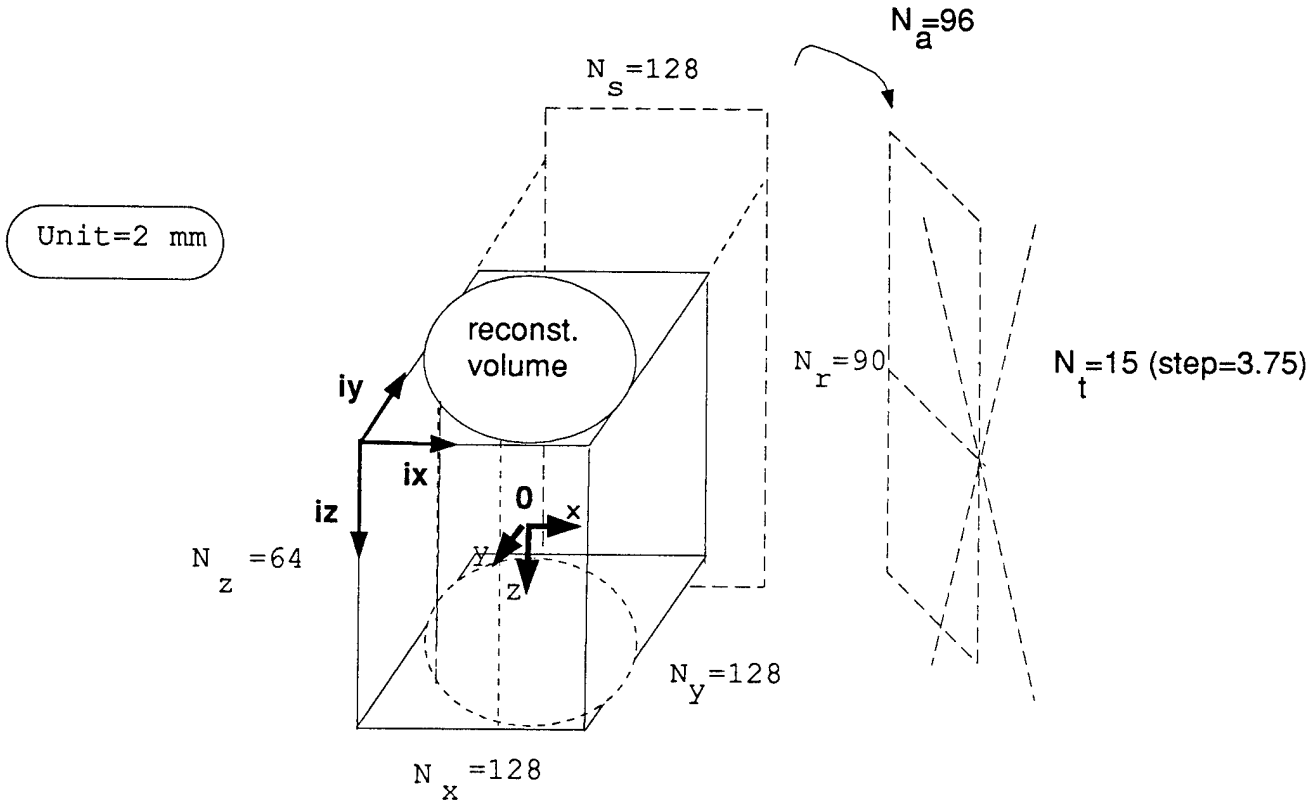


Figure 5. Diagram of reconstruction volume (cylinder), voxel space (cuboid) and panel of sensors

4.2. Voxel space

1. Consider a cuboid that tightly surrounds the cylinder which is the reconstruction volume.
2. This cuboid is divided into small cubic voxels of side VoxelSide=2mm.
3. Intensity range of values to be assigned to voxels: 16bits (short integer, 2B).
4. Thus we have:
   Nx=128 (columns, x-direction),
   Ny=128 (rows, y-direction),
   Nz=64 (slices, z-direction),
   and Nx×Ny×Nz=1M (requiring a total of 2MB to store).
5. The voxel at position (ix,iy,iz) is, using C notation:
   V[iz][iy][ix],
   where iz=0..(Nz-1), iy=0..(Ny-1), and ix=0..(Nx-1).
6. Correspondence with reconstruction volume (see figure 5, indices ix,iy,iz):
   The first slice corresponds to the top of 'reconstruction volume,' i.e., to the most negative z region. For each slice, the rows go in opposite direction in relation to y-axis (the first row starts at y=Dxy/2 mm), and the columns increase as x increases (the first column starts at x=-Dxy/2 mm). Thus, center of voxel V[0][0][0] is at (x,y,z)=(x(0), y(0), z(0)) mm, where
   x(0)=-(Dxy/2-VoxelSide/2) mm,
   y(0)=Dxy/2-VoxelSide/2 mm,
   z(0)=-Hz/2+VoxelSide/2 mm,
   and center of voxel V[iz][iy][ix] is at:
   x(ix)=(ix-Nx/2+0.5)×VoxelSide mm; ix=0..(Nx-1),
   y(iy)=(Ny/2-0.5-iy)×VoxelSide mm; iy=0..(Ny-1),
   z(iz)=(iz-Nz/2+0.5)×VoxelSide mm; iz=0..(Nz-1).
7. For display purpose, V[k][0][0] is located at upper left corner (in axial view).

4.3. Phantoms

1. Each phantom is contained within a large ellipsoid with its z-axis coinciding with the z-axis of xyz coordinate system and its center at (0,0,Hz/2) in mm. Between the planes z=Zmin=-Hz/2 mm and z=Zmax=Hz/2 mm, there are nObjects=69 spheres, distributed in three layers (A,B,C) (see section 3.1). These spheres contain the objects (features) that are ellipsoids. The activity is restricted to the reconstruction region(fig.1).
2. The radii of the large ellipsoid are:
   Rx=128 mm,
   Ry=96 mm,
   Rz=147.80 mm.
   The associated intensity is BKG (background). This value will be determined so that the total detected activity (attenuated data) from the large ellipsoid (background only) will be 80 million counts. Note: whole phantom is considered for attenuation, i.e, if a ray (tube) also intercepts part

of phantom without activity, line integral of activity will consider only the part of phantom with activity, while attenuation is computed on the basis of the whole intersection segment.

3. Each feature is inside a sphere, whose position and radius are described by (xcenter, ycenter, zcenter, sphereRadius) and whose background intensity is given by bkg (local additional background, usually equal to 0). See appendix D for list of sizes and locations of spheres.
4. Each feature (ellipsoid), has the following characteristics:
   a. same center as the enclosing sphere;
   b. radii (rx,ry,rz) are between 50% to 100% of sphereRadius, each independently randomly selected from a uniform distribution over this range;
   c. orientation: the ellipsoid above described is rotated by (in this order):
      ax (rotation around x-axis),
      ay (rotation around y-axis),
      az (rotation around z-axis),
      where each of these three angles is independently randomly chosen from a uniform distribution over the range $[0,\pi)$.
   d. activity is randomly assigned to be one of three equiprobable types: NORMAL (the activity in the feature is the same as in its background), HOT (the feature is a "hot spot"), or COLD (the feature is a "cold spot"). The activity assigned to hot spots (respectively, to cold spots) is higher (respectively, lower) than the activity in their background by the contrast listed in Table I.
   e. each feature contains at least 5 voxels inside and 5 voxels in feature background region (region between feature and enclosing sphere)
5. In order to accurately estimate the average intensity in a voxel in the phantom image, each voxel is subdivided into 64 sub-voxels (fig.6) and the intensity of each one of these is calculated as the sum of the intensities of all features and spheres which contain subvoxel center. The voxel intensity is the average of the intensities of its sub-voxels.

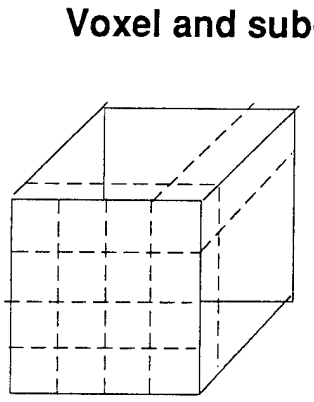


Figure 6. Illustration of voxel and subvoxels

6. Algorithm for Phantom3D creation (**.img**)
   a. Generate (randomly) feature parameters (size, orientation, activity)

b. Check for at least 5 voxels inside each feature and background region: if not, generate (randomly) new feature. We are considering "inside" if center and at least five vertices of voxel are inside specific region.

c. For each feature

- Get feature characteristics { rx,ry,rz, ax,ay,az, value }
- Update values of sub-voxels inside the sphere by feat.bkg (not needed if bkg=0)
- Update all sub-voxels inside this feature by feat.value

d. Calculate each voxel's value as average of the values of its sub-voxels

e. Store data (**.pde** and Petview format **.img**)

4.4. Projection data

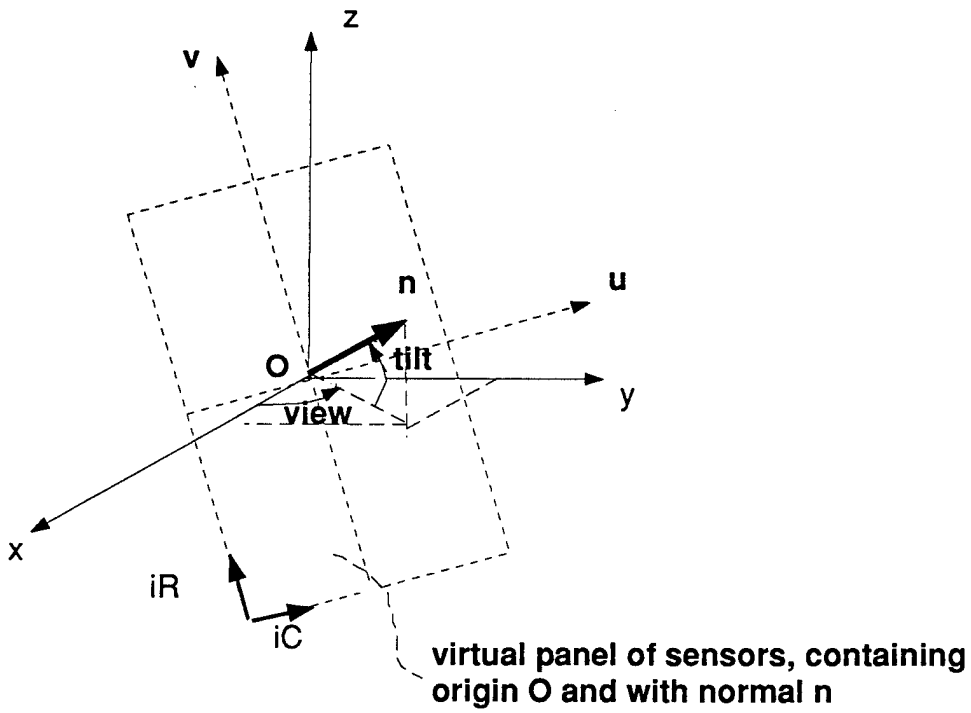


Figure 7. Coordinate system (u,v) of panel in relation to xyz-coordinate system of reconstruction volume. Vector n is normal to the panel.

1. It is assumed that data are rebinned into parallel raysums, and activity are restricted to the region between z=[-Hz/2,Hz/2] mm. We are considering all detectable rays for the given scanner geometry (cylinder):
   SCNdiameter=420mm,
   SCNheight=256mm.
   We are assuming that the center of phantom and reconstruction volume coincide with scanner center.

2. There is a rectangular matrix (of size nSCNrows×nSCNcols), called a *panel*, of sensors (SensorSide×SensorSide each), which is rotated to provide views and tilted to provide tilts (figure 5), with:
   SensorSide=2 mm,
   nSCNrows=90 (number of rows),
   nSCNcols=128 (number of columns),
   nSCNviews=96 (number of views, equally spaced between 0 and π),
   nSCNtilts=15 (number of tilts),
   TiltStep=3.75 (degrees).
   For each tilt and for each view, we calculate for each sensor a *raysum* and the collection of such raysums is referred to as the projection data.

   nSCNrows was calculated such that the panel can detect all rays detectable by the scanner for all considered tilts:

   a. For a given tilt 't', for all views of this tilt, the maximum region of detectable rays is limited by the projection of two circles: top (bottom) of scanner cylinder (limiting curve) and of reconstruction cylinder (expanding curve) (see figure 8).

   b. The limiting curve for all rays from scanner cylinder that can be detected is given by:

$$\frac{h_s}{2}cos(t) - sin(t).\sqrt{R_s^2 - u^2}$$

   c. Expanding curve for all possible rays for given tilt from reconstruction region (cylinder) is given by:

$$\frac{h_r}{2}cos(t) + sin(t).\sqrt{R_r^2 - u^2}$$

   where:
   h_s: height of scanner cylinder
   R_s: diameter of scanner cylinder
   h_r: height of reconstruction cylinder
   R_r: diameter of reconstruction cylinder
   t: tilting angle (t >=0)
   u: panel abscissa

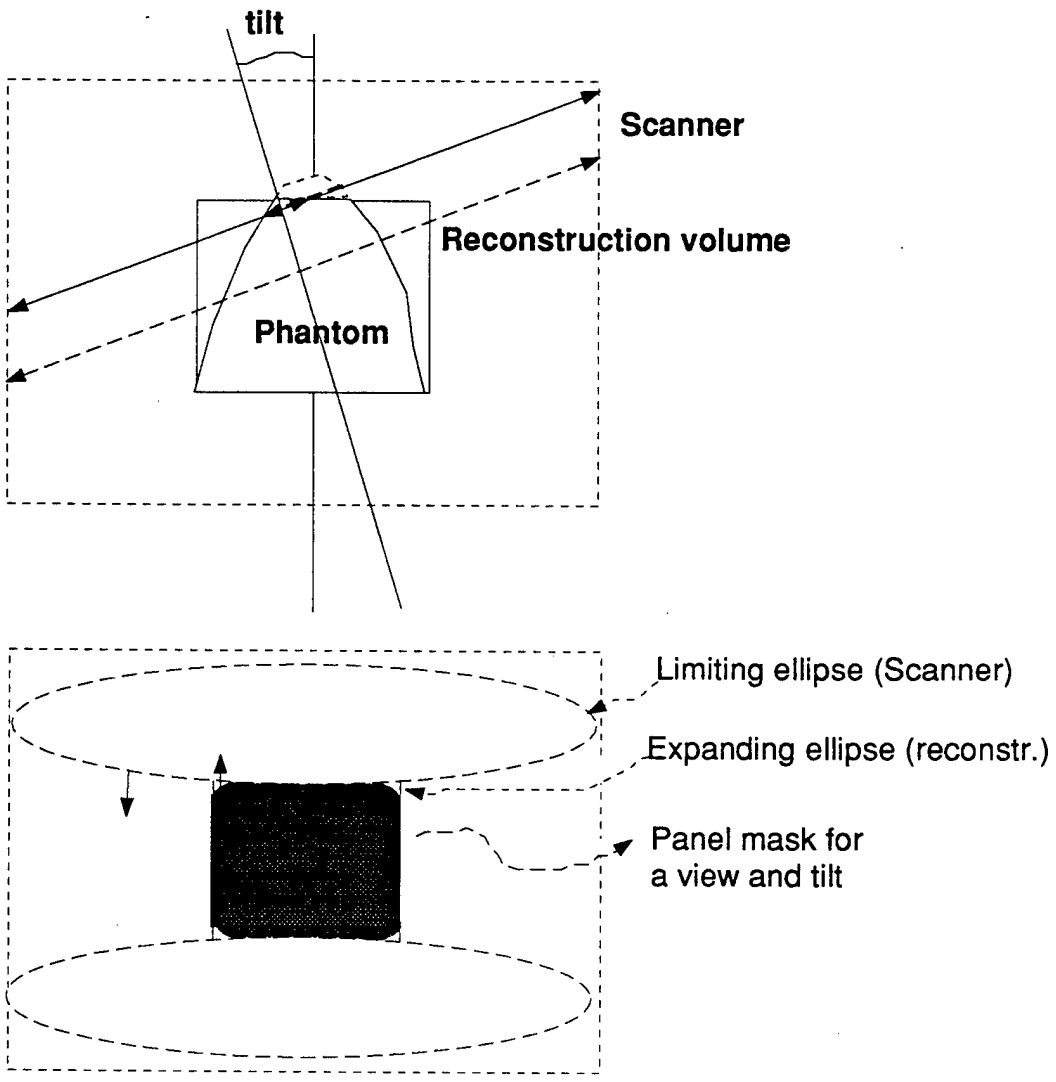


Figure 8. Illustration of panel mask for bins with detectable counts for a given tilt.

3. It is assumed that the center line of the panel is in plane z=0.

4. Rotation of the panel is around the z-axis (nSCNviews), using the right-hand rule direction. The first panel is at angle 0. (In case of no tilting, this corresponds to the yz plane.)

5. Tilting is around the middle line of the panel (figure 5), with nSCNtilts tilt positions. It is assumed nSCNtilts is odd. TiltStep=3.75 degrees. So the tilting varies from -int(nSCNtilts/2)×TiltStep to int(nSCNtilts/2)×TiltStep and index=0..(nSCNtilts-1). Thus index of 'no tilting' is int(nSCNtilts/2).

6. Calculation of raysums:
   First, raysums at finer resolution ('s' , 1mm apart) are calculated as accumulated line integrals

through features for each SensorSide/2 position. The effective raysum for a sensor 'p'(2 mm apart) will be the local average and sub-sampling of filtered raysums 's' (fig. 9),i.e:

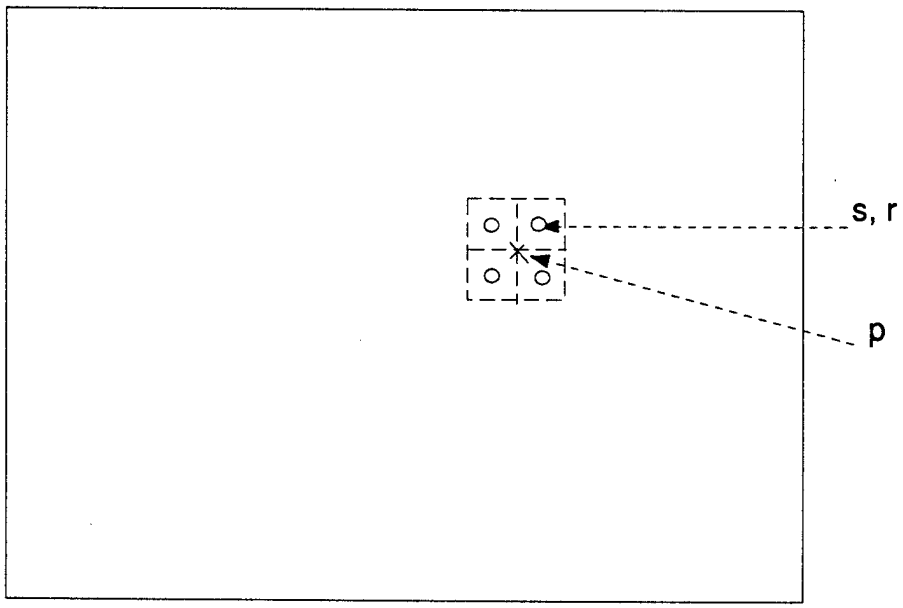


Figure 9. Illustration of panel sampling: raysum(p) and panel sub-sampling: sub-raysum (s,r)

a. Calculate 's'

Given a ray in 3D space, calculate the non-attenuated raysum through the given phantom, which is a set of ellipsoids.

b. Filter 's' to get 'r'

$$r_i = \frac{\sum_j w_j . s_{ij}}{\sum_j w_j}$$

where j: over local VALID neighbourhood

Filter (w) is radially symmetric with Gaussian shape of size 13x13 points and with FWHM=4 mm:

$$f(u,v) = \frac{1}{2\pi\sigma^2} * exp\left(-0.5\frac{u^2+v^2}{\sigma^2}\right)$$

$$\sigma = \frac{FWHM}{\left(2\sqrt{2ln2}\right)}$$

$$u, v = \{-6..6mm\}$$

Projection smooth is done to take into account inaccuracies in rebinning and in locating position of source (positron).

c. Local average and sub-sampling of 'r' yielding 'p' (fig. 9)

7. Noise (see section 3.2)
8. Store each panel data in Petview format (**.scn**) by rows p[iR], iR=0..nSCNrows-1 (fig. 7)

   Algorithm:

   For each panel

   a. Consider sub-raysums (SensorSide/2 by SensorSide/2)
   b. Reset all sub-raysums to zero
   c. For each feature=0..(nObjects-1) (including spheres and large ellipsoid),Do
      { Find all rays that pass through the feature;
      For each such ray, calculate the line integral (length of the segment defined by the intersection of the feature and the line, multiplied by the intensity) through the feature;
      Accumulate.
      }
   d. Filter (Gaussian)
   e. Calculate raysums as average of sub-raysums that form each raysum
   f. Noise generation
   g. Store data (Petview format)

Chapter 5. FILES AND FORMATS

5.1. Files

1. filename.pde: phantom and scanner geometry description file (described below)
2. filename.img: 3D-phantom data (Petview format)
3. filename.scn: 3D-phantom projection data (Petview format)
4. phaLengths.AttenF.scn: attenuation factor for each projection bin (Petview format). In our case, it contains, for each ray, $e^{-\mu L}$ where L is the intersection length (in mm) of the ray and the phantom, considered as a non-truncated ellipsoid and μ=0.0095 mm^{-1}.

5.2. Archiving format for .pde

A phantom and scanner description file (**.pde**, for generation of masks and for evaluation purposes) consists of a header in the format of the structure PHA_HDR, followed by nObjects=69 descriptions of features in the format of the structure FEATURE and structure USER_PARAM. Precise specifications of these structures are given in Appendix C.

5.3. Archiving format for .img and .scn

The phantom image data (**.img**) and projection data (**.scn**) are stored using Petview format (see Appendix E).

In order to make comparison simpler, we are normalizing background values to 1 "arbitrary property unit" per mm (relative counts/mm of detector "tube") and using a CT model for projection data. Therefore the expected values for voxels (q_j) in background region is 1 and if a ray intercepts 256 mm of background region with activity, the expected projection value (p_i) is 256. However, we can convert back to realistic units and to PET model using simple factors as shown below.

The parameter 'TotalCounts' provided by the user (defaults is 80 millions) refers to the total counts detected (attenuated projection data) by a scanner due to only background in phantom . Since we know the geometry of the phantom and scanner, and linear attenuation coefficient (μ=0.0095 mm^{-1}), we can obtain the relationship between 1 "arbitrary property unit" and counts/voxel and counts/bin. This factor is stored in file **.pde** as Param.BKGnew (0.292 for default parameters) and as pha_hdr.BKG.

$$Param.BKGnew = \frac{TotalCounts}{\sum_i L_i'.e^{-\mu.L_i}}$$

where:

L_i: intersection (in mm) of ray 'i' with entire phantom. (attenuation part)

L_i':intersection (in mm) of ray 'i' with phantom limited by reconstruction region (fig. 1) (active part of relative phantom).

$\sum_i$: sum over all possible (not missing) rays.

If we call :

r_j: value of voxel 'j' [relative counts/mm of "tube"]

p_i: value of projection 'i' [relative counts in bin]

L_{ij} : length of intersection of i-th ray with j-th voxel [mm]

CT reconstructin model tries to solve :

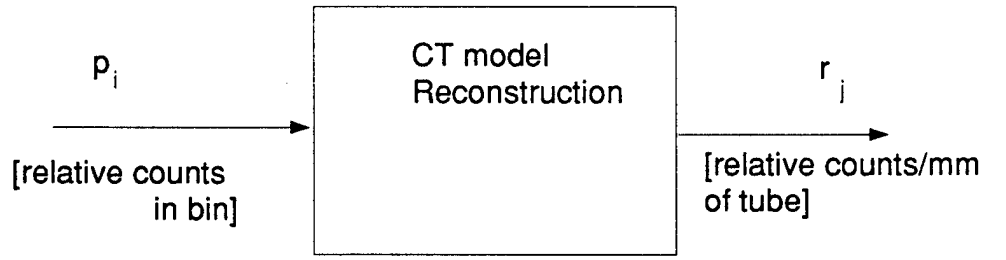
$$p_i = \sum_j L_{ij} . r_j$$

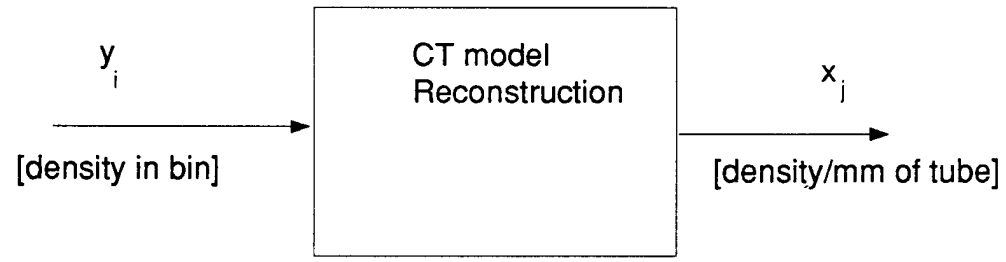
We will be referring to projection value (p_i, y_i, g_i) as the value of projection without attenuation, or if one prefers, projection value after attenuation correction.

If one wants to use data in CT model, but in realistic units per mm of "tube", it is enough to consider:

x_j = Param.BKGnew × r_j (for voxel) [density/mm of tube]

y_i = Param.BKGnew × p_i (for projections) [density in bin]





For PET model of projection data, we need to obtain the relationship between probability for a bin 'i' to detect a photon emitted at location 'j' (t_{ij}) and length of intersection of ray 'i' and voxel 'j' (L_{ij}).

Let :

f_j: number of photons emitted at location 'j' [counts in voxel]

g_i: number of photons detected at bin 'i' [counts in bin]

A: area of one detector [mm^2]

V: volume of one voxel [mm^3]

We have:

$$g_i = \sum_j t_{ij} . f_j$$

$$\sum_i t_{ij} = 1 \textit{ for all 'j' with no missing data}$$

Since $\sum_i L_{ij} \simeq \frac{V}{A}.nSCNtilts.nSCNviews = F\ [mm/voxel]$ (F=2880 for defaults parameters) we can approximate $t_{ij} \approx L_{ij}/F$

Then $g_i = \sum_j L_{ij}.\frac{f_j}{F}$. Comparing with equation $y_i = \sum_j L_{ij}.x_j$, we have for realistic CT model data:

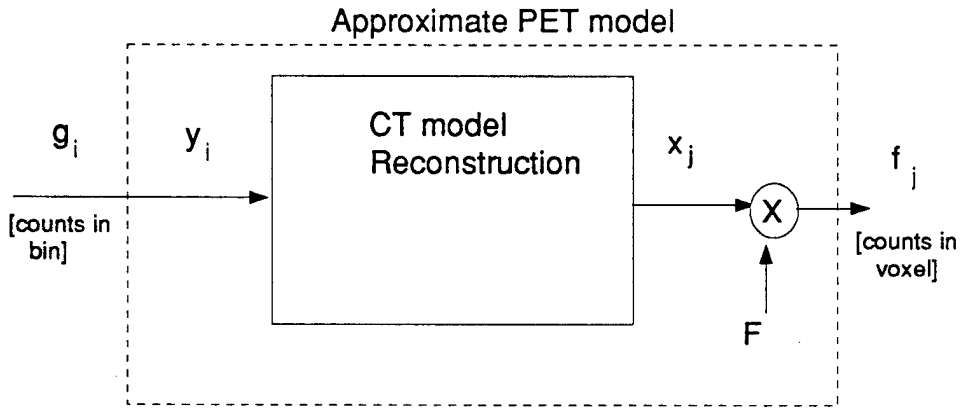
$g_i = y_i$

$f_j = F \times x_j$

Therefore, realistic values of PET model data can be approximated from the relative CT model data by:

f_j = Param.BKGnew × F × r_j (for voxel) [counts in voxel]

g_i = Param.BKGnew × p_i (for projections) [counts in bin]



Some possible scenarios for using our projection data and reconstruction programs:

1. Using our projection data p_i (relative CT data model) and provided reconstruction programs (for CT model)
   a. We will get reconstructed values r_j in relative values, i.e., the expected value in background is 1.
   b. If we want results in "density per mm of tube" we should multiply input or output data by Param.BKGnew, we get:
      x_j = Param.BKGnew × r_j
   c. If we want results in counts per voxel, we should multiply input or output data by Param.BKGnew × F, we get:
      f_j = Param.BKGnew × F × r_j
      (that is approximately what we would get if we have used a PET reconstruction model with projection data : g_i = Param.BKGnew × p_i)

2. Using our projection data (p_i) associated to a phantom (r_j) with other reconstruction programs:
   a. If it uses a CT model for reconstruction, it is enough converting our projection data (Petview format) to program's format (the result will be in relative units, r_j).
   b. If it uses a PET model for reconstruction, the reconstructed image will be: $F \times r_j$ [relative counts/voxel]. In order to obtain proper number of counts/voxel, we should multiply results by Param.BKGnew.
3. Using provided reconstruction programs (CT model) with other projection data:
   a. CT model projection data:
      - set Param.BKGnew=1
      - can be used directly and the background value of reconstructed image will be actual background value [density/mm (of tube)].
   b. PET model for projection data (g_i) associated to an image (f_j) [counts per voxel]:
      - set Param.BKGnew=1
      - the reconstructed image will be : $\frac{f_j}{F}$

Notes:

- In the noise generation discussed in previous sections, we used values of 'g_i'.
- Our data are stored in CT model —voxel values (r_j) and projection data (p_i). Before storing they are scaled (multiplied) by 'Param.Image_scale' and 'Param.MultFactorForStore', respectively. Default values of scaling factors are 1000. This procedure was necessary to overcome the equalization errors of transforming real number into integer, since Petview format (details in pet_mhdr and pet_sbhdr structures in file pet_hdr.h, Appendix E) for data is integer (2 bytes). In Petview write functions, the numbers in each slice are automatically scaled (by sh.imgscl and sh.scnscl, respectively) to not overflow integer type.

  More specifically, archiving is done:

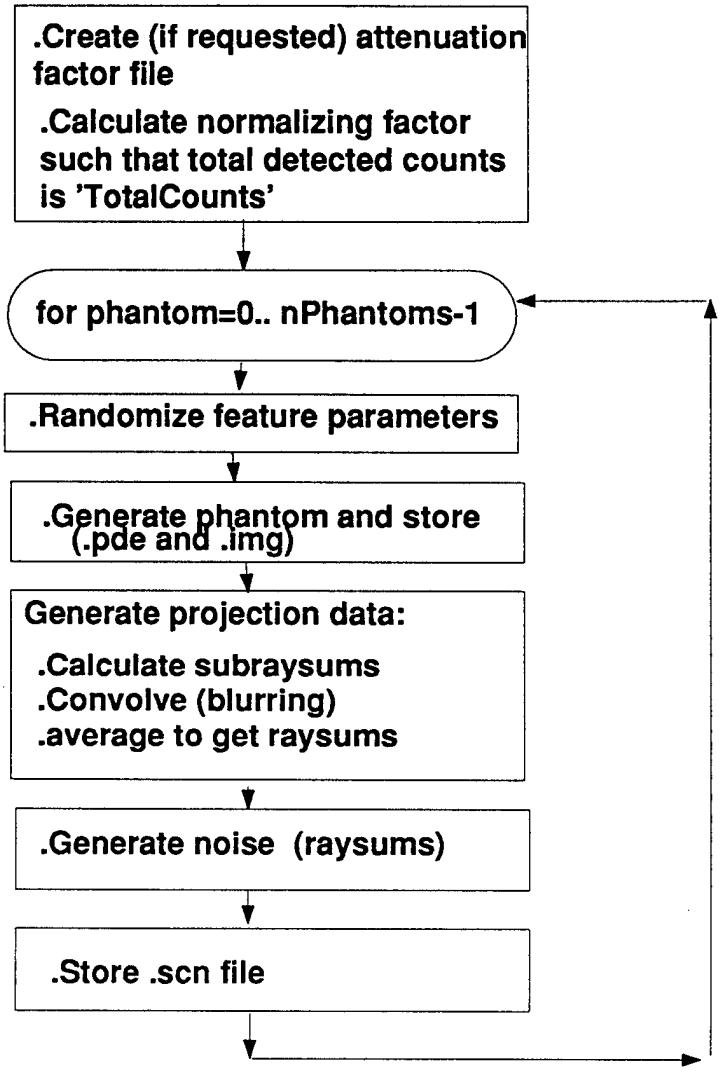
1. Phantom (**.img**)
   a. Storage is slice-oriented, usually from top to bottom of structure.
   b. For each slice (z fixed), the data are stored row by row, i.e.,
      for(iy=0; iy<Ny; iy++)
      for(ix=0; ix<Nx; ix++) store r[iz][iy][ix] × Param.Image_scale
      (Note: function wrimg() automatically calculates sh.imgscl and scales for each slice separately. So the stored value in file is r[iz][iy][ix] × Param.Image_scale/sh.imgscl, type int, where sh is a struct of type pet_sbhdr). However, Petview function rdimg() does not automatically rescale the read data.

   Size=Nz*Ny*Nx=1M (2MB)
2. Projection data (**.scn**):
   a *frame* parameter is a set of panels for a particular tilt (there are nSCNtilts frames),
   each frame has nSCNviews x nSCNrows×nSCNcols raysums;
   a *slice* parameter is a set of raysums for a particular view (panel), for a particular tilt (there are nSCNviews "slices" per frame),

each slice has nSCNrows × nSCNcols raysums;
C notation:
p[iT][iV][iR][iC]
iT=0..(nSCNtilts-1);
iV=0..(nSCNviews-1);
iR=0..(nSCNrows-1);
iC=0..(nSCNcols-1).

Algorithm:
For each frame, store slice by slice,
For each slice (panel), store row by row,i.e.:
store p[iT][iV][iR][iC] × Param.MultFactorForStore
where row start at most negative v (iR=0) (figure 7)
and column start at most negative u (iC=0).
(Note: function wrimg() automatically calculates sh.scnscl and scales for each slice separately. So the stored value in file is p[iT][iV][iR][iC] × Param.MultFactorForStore/sh.scnscl, type int, where sh is a struct of type pet_sbhdr). However, Petview function rdimg() does not automatically rescale the read data.
Size=nSCNtilts×nSCNrows×nSCNviews×nSCNcols=16.6M (33.2MB)

Chapter 6. FLOW DIAGRAM for genPhantom3d.c



Chapter 7. REFERENCES

Fiete R D, Barrett H H, Smith W E, and Myers K J 1987 Hotelling trace criterion and its correlation with human-observer performance. *J.Opt.Soc.Am. A* **4** 945–53

Fukunaga K 1972 *Introduction to Statistical Pattern Recognition* (New York: Academic Press)

Furuie S S, Herman G T, Narayan T K, Kinahan P, Karp J S, Lewitt R M, Matej S 1994 A methodology for testing for statistically significant differences between fully 3D PET reconstruction algorithms. *Physics in Medicine and Biology*

Gooley T A and Barrett H H 1992 Evaluation of statistical methods of image reconstruction through ROC analysis. *IEEE Trans.Med.Imag.* **MI–11** 276–83

Green D M and Swets J A 1988 *Signal Detection Theory and Psychophysics* (Los Altos, CA: Peninsula Publishing, reprint edition)

Hanson K M 1990 Method of evaluating image-recovery algorithms based on task performance. *J.Opt.Soc.Am.A* **7** 1294–304

Herman G T 1980 *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography* (New York: Academic Press)

Herman G T and Odhner D 1991 Performance evaluation of an iterative image reconstruction algorithm for positron emission tomography. *IEEE Trans.Med.Imag.* **MI–10** 336–46

Herman G T and Yeung K T D 1989 Evaluators of image reconstruction algorithms. *Internat.J.Imag.Syst.Techn.* **1** 187–95

Herman G T and Meyer L B 1993 Algebraic reconstruction techniques can be made computationally efficient. *IEEE Trans.Med.Imag.* **MI–12** (3):600–9, Sep 93.

Kinahan P E and Rogers J G 1989 Analytic 3D image reconstruction using all detected events. IEEE Trans.Nucl.Sci. **NS-36** 964–8

Llacer J, Veklerov E, Baxter L R, Grafton S T, Griffeth L K, Hawkins R A, Hoh C K, Mazziotta J C, Hoffman E J, and Metz C E 1993 Results of a clinical operating characteristic study comparing filtered backprojection and maximum likelihood estimator images in FDG PET studies. *J.Nucl.Med.* **34** 1198–203

Lewitt R M 1990 Multidimensional digital image representation using generalized Kaiser-Bessel window functions. *J.Opt.Soc.Amer.A* **7** 1834–46

Lewitt R M 1992 Alternatives to voxels for image representation in iterative algorithms. *Phys.Med.Biol.* **37** 705–16

Matej S and Lewitt R M 1992 Image representation and tomographic reconstruction using spherically-symmetric elements. *Proc. 1992 Nucl.Sci.Symp. Med. Imag.Conf.* (Orlando, Florida:IEEE) 1191–3

Matej S, Herman G T, Narayan T K, Furuie S S, Lewitt R M, and Kinahan P 1994 Evaluation of task-oriented performance of several fully 3–D PET reconstruction algorithms. *Phys.Med.Biol.*

Meyer S L 1975 *Data Analysis for Scientists and Engineers* (New York: John Wiley & Sons) pp 355–8

Mould R F 1989 *Introduction to Medical Statistics* (Bristol, England: Adam Hilger, 2nd edition)

Rowe R W and Dai S 1992 A pseudo-Poisson noise model for simulation of positron emission tomographic data. *Med.Phys.* **19** 1113–9

Swets J A and Pickett R M 1982 *Evaluation of Diagnostic Systems: Methods from Signal Detection Theory* (New York: Academic Press)

Udupa J K and Herman G T (editors) 1991 *3D Imaging in Medicine* (Boca Raton, FL: CRC Press)

Wang H, Jaszczak R J and Coleman R E 1992 Solid geometry-based object model for Monte Carlo simulated emission and transmission tomographic imaging systems. *IEEE Trans.Med.Imag.* **MI–11** 361–72

Yeung K T D and Herman G T 1989 Objective measures to evaluate the performance of reconstruction algorithms. *SPIE* **vol. 1092** 326–35

Appendix A Justification for contrast function

A common index for measuring detectability, when frequency distribution for a given decision variable is nearly Gaussian, is *detection signal-to-noise ratio* given by (Fukunaga, 1972; Hanson, 1990):

$$d = \frac{\overline{x_1} - \overline{x_0}}{\sqrt{\frac{\sigma_1^2 + \sigma_0^2}{2}}}$$

where $\overline{x_1}, \sigma_1^2, \overline{x_0}, \sigma_0^2$, are average and variance of signals x_1 (feature) and x_0 (background) respectively.

For reconstructed images, it can be assumed that the noise variance of voxel intensities inside features and outside are equal and independent of the size of features. Then,

$$d = \frac{\overline{x_1} - \overline{x_0}}{\sigma}$$

where $\sigma^2 = \sigma_1^2 = \sigma_0^2$ (constant over whole phantom).

For a given feature f with contrast c_f, we have:

$$E\left(x_1\right) - E\left(x_0\right) = c_f$$

Then,

$$E\left(d_f\right) = \frac{E(\overline{x_1}) - E(\overline{x_0})}{\sigma} = \frac{E(x_1) - E(x_0)}{\sigma} = \frac{c_f}{\sigma}$$

and

$$Var\left(d_f\right) = \frac{Var(\overline{x_1}) + Var(\overline{x_0})}{\sigma^2} = \frac{\frac{\sigma^2}{n_1} + \frac{\sigma^2}{n_0}}{\sigma^2} = \frac{n_0 + n_1}{n_0 . n_1}$$

where we assumed that the noise variance is constant $\sigma^2 = \sigma_1^2 = \sigma_0^2$ and n_0, n_1 are number of voxels considered for averaging.

In the analysis of detectability, for a given reconstruction, we are using phantom features with different sizes. Therefore, it is important to have approximately the same detection S/N ratio $\frac{E(h_f)}{\sqrt{Var(h_f)}}$ for all the feature sizes. This can be accomplished by setting the contrast of features proportional to their sizes:

$$c_f \propto \sqrt{\frac{n_0 + n_1}{n_0 . n_1}} \tag{23}$$

For our case, the feature is inside a sphere that provides background region for the feature. Thus, the number of voxels is proportional to R^3 and consequently $c_f \propto R^{-\frac{3}{2}}$. However due to other effects (such as border effect, randomness of sizes) experimentally we found that $c_f \propto R^{-2}$ yields comparable detectability for different sizes. In our experiment we considered 3 ranges of enclosing spheres sizes ([4.0 mm, 6.4 mm], [8.0 mm,10.0mm], [16.0 mm, 20.0mm]) and we used the function $c_f = \frac{12}{R^2}$

Appendix B Justification for detectability index formulae

In Appendix A we have derived a theoretical formula for the feature contrast, enabling comparable detectability for features of different sizes. Since we are going to use ROC analysis with features with different sizes, it is important to have also $E\left(d_f\right)$ and $Var\left(d_f\right)$ approximately independent of the size. It can be achieved by scaling detectability index d_f by contrast function.

A new detectability index can be defined by:

$$h_f = \frac{d_f}{c_f}$$

Considering contrast function (Appendix A) as:

$$c_f = k.\sqrt{\frac{n_0 + n_1}{n_0.n_1}}$$

Then $E\left(h_f\right)$ and $Var\left(h_f\right)$ will be independent of the feature size:

$$\mathrm{E}\left(h_f\right) = \frac{1}{\sigma}$$
$$Var\left(h_f\right) = \frac{1}{c_f^2}\left(\frac{1}{n_1} + \frac{1}{n_0}\right)$$

We have (k=1):

$$Var\left(h_f\right) = 1$$
$$h_f = \sqrt{\frac{n_0.n_1}{n_0 + n_1}}.\frac{\left(\overline{x_1} - \overline{x_0}\right)}{\sigma}$$

One estimate of σ is (Meyer, 1975):

$$\sigma^2 \cong \frac{n_0.v_0^2 + n_1.v_1^2}{n_0 + n_1},$$

where $v_0^2 = \frac{1}{n_0}.\sum\left(x_{0i} - \overline{x_0}\right)^2$ and $v_1^2 = \frac{1}{n_1}.\sum\left(x_{1i} - \overline{x_1}\right)^2$.

Hence we have equation (4):

$$h_f = \sqrt{n_0.n_1}.\frac{\left(\overline{x_1} - \overline{x_0}\right)}{\sqrt{n_0.v_0^2 + n_1.v_1^2}}$$

Appendix C Definitions (file:genPhantom3d.inc)

File: genPhantom3d.inc

```
/*-------------------------------------------------------
Definitions for reconstruction volume and voxel space
   -------------------------------------------------------*/
#define _Dxy 256 /* mm, diameter of reconstr.cylinder */
#define _Hz 128 /* mm, height of reconstr. cylinder */
#define _VoxelSide 2 /* mm */
#define _nSubVoxelsPerSide 4 /* 64 sub-voxels per Voxel */
#define _nObjects 69
#define _Nx (_Dxy+_VoxelSide-1)/_VoxelSide
#define _Ny _Nx
#define _Nz (_Hz+_VoxelSide-1)/_VoxelSide
/*-------------------------------------------------------
Definitions for truncated ellipsoid and features
   -------------------------------------------------------*/
#define _Zmin - _Hz/2
#define _Zmax _Hz/2
#define _Rx 128.0 /* mm, radius in x-axis */
#define _Ry 96.0
#define _Rz 147.80
#define _BKG 1.00  /* Initial relative value .Later it is set equal to Param.BKGnew*/
#define _hA 60.0 /* mm, height of region A */
#define _hB 40.0
#define _hC 28.0
#define _RA 20.0 /* mm, radius of large spheres in A */
#define _RB 16.0 /* mm, radius of large spheres in B */
#define _RC 10.0 /* mm, radius of large spheres in C */
#define _rA 8.0 /* mm, radius of small spheres in A */
#define _rB 6.4 /* mm, radius of small spheres in B */
#define _rC 4.0 /* mm, radius of small spheres in C */
#define _MinNumVoxels  5  /* minimum feature size  */
#define _MinNumVertices 5 /* minimum feature size */
#define _MaxNumLoops 300 /* for randomizing parameters */
/*-------------------------------------------------------
Definitions for projection data
   -------------------------------------------------------*/
#define _SCNdiameter 420  /* mm */
#define _SCNheight 256  /* mm */
#define _SensorSide 2 /* mm */
#define _nSCNrows 90
#define _nSCNcols 128 /* num.of sensor cols in the scanner */
#define _nSCNviews 96 /* num.of views around the phantom [0..pi) */
#define _nSCNtilts 15 /* num.of tilts */
#define _TiltStep 360.0/_nSCNviews /* 3.75 degrees */
#define _TotalCounts 80.0e6 /* total num.of emissions detected */
#define _nSubSensorPerSide 2 /* 4 sub-projections per projection */
#define _ATTENcoef 0.0095  /* attenuation coef.  in mm-1  */
#define _coefProjPoisson 0.0  /* coef.influence of raysum onto pseudoPoisson*/
#define _FLTdiameter 13.0 /* mm, filter diameter  */
#define _FWHM  4.0 /* mm, of filter  */
#define _MultFactorForStore  1000.0 /* multiplier for storing .scn*/
```

```
#define _Image_scale 1000.0 /* multiplier for storing .img */
#define _MissingData 1 /* true missing data */
/*-------------------------------------------------------
Types and structure definitions
    -------------------------------------------------------*/
typedef
struct PHA_HDR
{
/* ---------------------------------------------
Header (256 B)
    ---------------------------------------------*/
short nObjects; /* # of objects */
/* body: truncated ellipsoid. Center at (0,0,-Hz/2) */
float Rx,Ry,Rz; /*radii in mm */
float BKG; /* intensity inside truncated ellips.*/
float  ATTENcoef;  /* attenuation of main body (background*/
/* Reconstruction volume */
float Zmin; /* top of phantom,usually -Hz/2 mm */
float Zmax; /* bottom of phantom, usually Hz/2 mm */
float Xmin, Xmax;
float Ymin, Ymax;
short Nx,Ny,Nz; /* num.  of elements in each direction */
char comments[204 ];
} pha_hdr ;

typedef struct FEATURE
{
/* ---------------------------------------------
Data (16*4=64B)
    ---------------------------------------------*/
/* ** object ** */
float sphereRadius;
float xcenter,ycenter, zcenter;
float bkg; /* aditional intensity inside sphere */
float rx,ry,rz; /* radii of ellipsoid */
float ax,ay,az; /* rotation of feature (radians)
    in relation to x,y,z axis*/
float value; /* aditional intensity inside feature*/
float  AttenCoef;  /* atten.coef.of feature */
float spare0,spare1,spare2; /* future use */
} feature;

typedef struct USER_PARAM
 {
  char prefix[30];
 BOOLEAN RANDOM;
 int nPhantoms;
 BOOLEAN POISSON;
 BOOLEAN  FOV;
 BOOLEAN  INTERSECTIONS;
 char  fileLengths[80];
 float  coefProjPoisson;
 float TotalCounts;
 char  comments[200];
```

```
/* for reconstruction volume and voxel space */
float Dxy,
  Hz,
  VoxelSide;
int   nSubVoxelsPerSide,
  nObjects,
  Nx,
  Ny,
  Nz;
float Zmin,
  Zmax;
/* for truncated ellipsoid and features
BKGnew: actual value of background
*/
float Rx,
  Ry,
  Rz,
  BKG, BKGnew,
  ATTENcoef,
  hA,
  hB,
  hC,
  RA,
  RB,
  RC,
  rA,
  rB,
  rC;
int  MinNumVoxels;
/* for projection data */
float SensorSide;
int  SCNdiameter,
  SCNheight,
  nSCNrows,
  nSCNcols,
  nSCNviews,
  nSCNtilts,
  nSubSensorPerSide;
float TiltStep;
float  FWHM;
float  FLTdiameter;
float  MultFactorForStore;
float Image_scale;
int MissingData;
char Corname[80]; /* name of correction file*/
/* area under program control(temporary mem) */
char  phantomN[4]; /* phantom # 0..999 */
float  factorSumR1;
long  seed;  /* for numb.random*/
int  TESTING; /* flag for test */
} UserParam;
```

Appendix D Table of sphere sizes and locations

Feature	Sphere radius (mm)	(x,y,z) of sphere center (mm)
1	20.0	(-40.00, 40.00, 34.00)
2	20.0	(0.00, 40.00, 34.00)
3	20.0	(40.00, 40.00, 34.00)
4	20.0	(-80.00, 0.00, 34.00)
5	20.0	(-40.00, 0.00, 34.00)
6	20.0	(0.00, 0.00, 34.00)
7	20.0	(40.00, 0.00, 34.00)
8	20.0	(80.00, 0.00, 34.00)
9	20.0	(-40.00,-40.00, 34.00)
10	20.0	(0.00,-40.00, 34.00)
11	20.0	(40.00,-40.00, 34.00)
12	8.0	(-60.00, 20.00, 50.00)
13	8.0	(60.00, 20.00, 50.00)
14	8.0	(-60.00,-20.00, 50.00)
15	8.0	(60.00,-20.00, 50.00)
16	8.0	(-60.00, 20.00, 34.00)
17	8.0	(60.00, 20.00, 34.00)
18	8.0	(-60.00,-20.00, 34.00)
19	8.0	(60.00,-20.00, 34.00)
20	8.0	(-60.00, 20.00, 18.00)
21	8.0	(60.00, 20.00, 18.00)
22	8.0	(-60.00,-20.00, 18.00)
23	8.0	(60.00,-20.00, 18.00)
24	16.0	(-32.00, 32.00,-16.00)
25	16.0	(0.00, 32.00,-16.00)
26	16.0	(32.00, 32.00,-16.00)
27	16.0	(-64.00, 0.00,-16.00)
28	16.0	(-32.00, 0.00,-16.00)
29	16.0	(0.00, 0.00,-16.00)
30	16.0	(32.00, 0.00,-16.00)
31	16.0	(64.00, 0.00,-16.00)
32	16.0	(-32.00,-32.00,-16.00)
33	16.0	(0.00,-32.00,-16.00)
34	16.0	(32.00,-32.00,-16.00)
35	6.4	(-48.00, 16.00, -3.20)
36	6.4	(48.00, 16.00, -3.20)

37	6.4	(-48.00,-16.00, -3.20)
38	6.4	(48.00,-16.00, -3.20)
39	6.4	(-48.00, 16.00,-16.00)
40	6.4	(48.00, 16.00,-16.00)
41	6.4	(-48.00,-16.00,-16.00)
42	6.4	(48.00,-16.00,-16.00)
43	6.4	(-48.00, 16.00,-28.80)
44	6.4	(48.00, 16.00,-28.80)
45	6.4	(-48.00,-16.00,-28.80)
46	6.4	(48.00,-16.00,-28.80)
47	10.0	(-20.00, 20.00,-50.00)
48	10.0	(0.00, 20.00,-50.00)
49	10.0	(20.00, 20.00,-50.00)
50	10.0	(-40.00, 0.00,-50.00)
51	10.0	(-20.00, 0.00,-50.00)
52	10.0	(0.00, 0.00,-50.00)
53	10.0	(20.00, 0.00,-50.00)
54	10.0	(40.00, 0.00,-50.00)
55	10.0	(-20.00,-20.00,-50.00)
56	10.0	(0.00,-20.00,-50.00)
57	10.0	(20.00,-20.00,-50.00)
58	4.0	(-30.00, 10.00,-42.00)
59	4.0	(30.00, 10.00,-42.00)
60	4.0	(-30.00,-10.00,-42.00)
61	4.0	(30.00,-10.00,-42.00)
62	4.0	(-30.00, 10.00,-50.00)
63	4.0	(30.00, 10.00,-50.00)
64	4.0	(-30.00,-10.00,-50.00)
65	4.0	(30.00,-10.00,-50.00)
66	4.0	(-30.00, 10.00,-58.00)
67	4.0	(30.00, 10.00,-58.00)
68	4.0	(-30.00,-10.00,-58.00)
69	4.0	(30.00,-10.00,-58.00)

Appendix E Petview format

The data files for image and projection data are stored as *.img* and *.scn* files, using *Petview* format, whose header (*pet_mhdr*) and subheader (*pet_sbhdr*) structure are described below.

In order to read and write files in *Petview* format, the following routines are needed, which are include in file *imagio1.1.c (imagio.h)*. Please see documentation in these files for further information. The included reconstruction algorithms may also be useful as examples for reading/writing data in *Petview* format.

Routines for reading:

1. fileopn() : opens file
2. getmhdr() : reads main header
3. getsbhdr() : reads sub-header
4. rdimg() : reads data (1 slice)
5. filclose() : closes file

Routines for writing:

1. filcre() : creates file
2. wrmhdr() : writes main header
3. wrimg() : writes data (1 slice)
4. filclose() : closes file

The following routines that are included in file *MIPGphaFunc.c* may be useful :

1. mh_init() : initializes main header
2. sh_init() : initializes sub-header
3. mh_shIMGupdate(): updates main header and sub_header structure of image data based on PARAM structure content
4. mh_shSCNupdate(): updates main header and sub_header structure of projection data based on PARAM structure content

```
/* .....File : pet_hdr.h .....
Purpose: Contains main header and sub header definitions. It should be used as an include file in programs which use
the imagio routines to read/write the headers.
By: Petview/UGM group
Date: Revised July 9, 1991
*/
/** C1 - Comment #1 - eliminate the setting of these parms by UNIX localtime() call in wrmhdr(). The application should
set these values only, not wrmhdr().
C2 - replacing the 'ybias' parameter name with 'sclsep', Words 157-158 UGM scanner physical slice separation is 1.0
mm, but for older University scanner, the separation is 2.0 mm. Currently: if ('ybias' == 1.0) data acquired on UGM
scanner else data acquired on University scanner
C3 - scntype (scan type) contains all possible .img types and .scn types
**/
struct pet_mhdr {
short filtyp; /* File type. 1=.SCN, 2=.IMG, 3=.VID 4=.OTHER */ /*C1*/
```

```
short daycre; /* Day of file creation.*/
/*C1*/ short mocre; /* Month of file creation.*/
/*C1*/ short yrcre; /* Year of file creation.*/
/*C1*/ short hrcre; /* Hour of file creation.*/
/*C1*/ short mincre; /* Minute of file creation.*/
/*C1*/ short seccre; /* Second of file creation.*/
short pattyp; /* Scan type, 0=test, 1=patient. */
short shdtyp; /* Subheader type, 0=UCLA/CTI, 1=UGM. */
short duratn; /* Duration of scan in seconds. */
/*C3*/ short scntyp; /* 0=not a scan, 1=blnk, 2=trmiss, 3=emiss 4=norm 5=obl, 6=bsi */
short numray; /* Number of rays per projection. */
short numang; /* Number of angles in collected data. */
short slcthk; /* Slice thickness in mm. */
short isotop; /* Isotope (1=F18, 2=O15, 3=C11 4=Ga-68, 5=N-13, 6=Rb-82, 7=Cu-62) */
float slope; /* Calibration slope, nci/ml/cts/pix/min. */
float intcpt; /* Calibration intercept, nci/ml. default=0 */
short injtim; /* Injection time offset for scan start. */
short nslice; /* Number of slices per frame. */
short nframe; /* Number of frames collected. */
char bthday[7]; /* Subject birthdate, (MMDDYY).*/
char ssn[10]; /* Subject social security number.*/
short petnum; /* PET number.*/
float dmax; /* Diameter of region being imaged.*/
float angmax; /* Maximum acceptance angle.*/
float x0; /* X-coord. of center of image.*/
float y0; /* Y-coord. of center of image.*/
float z0; /* Z-coord. of center of image.*/
float nevent; /* Number of raw events processed.*/
float nsino; /* Number of events stored in sinogram.*/
short eglob_low; /* Global energy - upper threshold */
short eglob_up; /* Global energy - upper threshold */
short eloc_low; /* Local energy - lower threshold */
short eloc_up; /* Local energy - upper threshold */
short orientation; /* 0 - not applicable, 1=Head First, 2=Feet First*/
char scan_swrel[6]; /* scanner software release; range 01.00-99.99 */
/*C2*/ float slcsep; /* Physical Slice Separation 1 = UGM */
char fctrfil[20]; /* Factor file name. (PMT gains).*/ /* was 16 */
char baselin[20]; /* Baseline file name. (DC offsets).*/ /* was 16 */
char dstpkfl[20]; /* Distortion peak file name.*/
char aqprotocol_name[20]; /* acquisition protocol name */
short aqprotocol_type; /* 1=Emiss-Static, 2=Emiss-Dynamic, 3=Trans, 4=Gated Cardiac, 5=Whole Body */
char patient_name[30]; /* patient name */
float reslice_ang1; /* Reslicing (OBL) angle 1 */
float reslice_ang2; /* Reslicing (OBL) angle 2 */
float reslice_ang3; /* Reslicing (OBL) angle 3 */
short minslc; /* Minimum Slice number */
short maxslc; /* Maximum Slice number */
short minfrm; /* Minimum Frame number */
short maxfrm; /* Maximum Frame number */
short trailexists; /* 1= trailer exists, 0 = no trailer */
long trailbeg; /* unsigned 32 bit number = # of bytes from file beginning indicating where the trailer begins */
}; /* end pet_mhdr */
/*****************************************************************/
struct pet_sbhdr {
/* currently wrsbhdr does not write the following subheader parameter: gatint */
```

```
short datype; /* Data type (1=byte,2=I*2,4=R*4orI*4,8=R*8) */
short xdim; /* X dimension (128 for .IMG, 256 or 128 for .SCN) */
short ydim; /* Y dimension (128 for .IMG, 192 or 96 for .SCN) */
short slcnum; /* Slice number.*/
short gatint; /* Gating interval (msec). */
float magfac; /* Reconstruction magnification (nominally 1) */
/* magfac used in analyze anz2u.c in imagio.c Convert UCLA to Analyze Format */
float imgscl; /* Image scale factor (nominally 1.) */
int imgmin; /* Minimum value in image; scaled to short by wrshdr()*/
int imgmax; /* Maximum value in image; scaled to short by wrshdr()*/
float scnscl; /* Sinogram scale factor */
short strhr; /* Starting time of this frame: hour.*/
short strmin; /* Starting time of this frame: minute.*/
short strsec; /* Starting time of this frame: second.*/
int scnmin; /* Minimum value in sinogram; scaled to short by wrshdr()*/
int scnmax; /* Maximum value in sinogram; scaled to short by wrshdr()*/
short endhr; /* Ending time of this frame: hour.*/
short endmin; /* Ending time of this frame: minute.*/
short endsec; /* Ending time of this frame: second.*/
long midtim; /* Time from start of scan to frame midtime of frame. */
/* midtim used by cmrglu.c */
short scnlen; /* Scan duration in seconds.*/
float imgsum; /* Total number of events in image.*/
float scnsum; /* Total number of events in sinogram.*/
/* additional parameters to add to the subheader: */
/* Space available from words 113 - 256 */
float bgsngrt; /* Beginning singles rate - all detectors */
float bgcoincrt; /* Beginning coincidence rate - all bankpairs */
float endsngrt; /* Ending singles rate - all detectors */
float endcoincrt;/* Ending coincidence rate - all bankpairs */
float deadtimefac;/* Deadtime Correction Factor default=1.0 */
short bedpos; /* Bed position for whole body scanning */
RECONINFO reconinfo;
}; /* end pet_sbhdr */ /*************************************************************************/
typedef struct reconinfo{
/**** Reconstruction Parameters *****/
/* General */
char recon_swrel[6]; /* reconstruction software release; range 01.00-99.99 */
char analy_swrel[6]; /* display/analysis(petview) software release; range 01.00-99.99 */
char recprotocol_name[20]; /* reconstruction protocol name */
char insinofile[20]; /* sinogram file to be reconstructed */
short slc_add; /* slice addition 1=Yes 0=No */
short slc_space; /* if slice add, Slice Spacing in slices */
short slc_thick; /* if slice add, Slice Thickness in slices */
short frame_add; /* frame addition 1=Yes 0=No */
short frame_space; /* if frame add, Frame Spacing in frames */
short frame_thick; /* if frame add, Frame Thickness in frames */
/* Filtered Backprojection */
short fltr_type; /* 0=NONE, 1=HANNING, 2=GAUSSIAN */
short smoth; /* Smooth Factor for Hanning Filter */
/* Background Subtraction */
short bgsub_type; /* 0=NONE, 1=UNIFORM, 2=NONUNIFORM */
short edge_exp; /* Number of bins to expand edge */
short ang_avg; /* Number of angles to average */
float bck_coeff; /* Non-uniform background coefficient */
```

```
short bck_wid; /* Number of bins for fitting tails radially */
/* Attenuation Correction */
short attncor_type; /* 0=NONE, 1=TRAN, 2=ELLIPSE, 3=READ */
float attn_coef; /* Attenuation Coefficient in 1./cm. */
char regfile[20]; /* Attenuation region file name */
char proc_transinofile[20]; /* Processed Transmission sinogram file of attenuation coefficients */
float skull_comp; /* Ad hoc correction for effect of skull */
/* Normalization */
short norm_type; /* 0=NONE, 1=AXIAL, 2=EFFICIENCY, 3=BOTH */
short smp_norm; /* Sampling Pattern Removal 0=No 1=Yes */
char axnfile[20]; /* Axial Normalization file name */
char effnormfile[20]; /* Efficiency Normalization file name */
/* Gap Compensation */
short gap_comp; /* gap compensation 1=Yes 0=No */
short gap_expand; /* Number of bins to expand gaps. */
short num_iter; /* Number of iterations. */
short gap_dir; /* Gap insertion direction. (1=HORIZONTAL 2=VERTICAL). */
short gap_edge; /* Use edge information (1=TRUE 0=FALSE).*/
short gap_fthr; /* Width of sinogram gap feathering. */
short mod_iter; /* Modified last iteration. (1=TRUE 0=FALSE)*/
float freq_mask_fwhm; /* FWHM of freq mask smoothing filter. */
/* Decay Correction: */
short decay_corr; /* Decay correction (1=TRUE 0=FALSE). */
/* Transmission scan processing. */
char transinofile[20];/* Transmission scan sinogram. */
char blnksinofile[20];/* Blank scan sinogram. */
float tran_ray_fwhm;/* FWHM of Gaussian used for transverse smoothing.*/
float tran_axl_fwhm;/* FWHM of Gaussian used for axial smoothing.*/
short surv_mask; /* Number of rays to mask survival prob to 1.*/
} RECONINFO;
```

Appendix F Script samples

File: **buildall.bat**

```
#! /bin/csh
# file: buildall.bat Sergio Furuie MIPG 22Sep93
#
# This script builds all programs related to evaluation of 3d PET
# It should be run in subdirectory 'bin'
#
# Usage for example for sparc:
# buildall.bat sparc
# (In this case, it creates a subdirectory called 'sparc')
# It assumes:
# 1) environment variable 'dir3d' set. Example: setenv dir3d /usr/EVAL3DPET
# 2) Sources have been copied to subdirectories of 'sources':
# 'bib', 'gen', 'rec', 'eval', 'util'
# Machine dependent
setenv dirMach $1
mkdir $dirMach
# Machine independent.
set sourcesDir=$dir3d/sources
set genDir=$sourcesDir/gen
set bibDir=$sourcesDir/bib
set recDir=$sourcesDir/rec
set evalDir=$sourcesDir/eval
set trainDir=$sourcesDir/train
set utilDir=$sourcesDir/util
# creating library
cd bibDir
make -f bib.mak
\rm *.o
# Phantom generation
cd genDir
make -f gen.mak
\rm *.o
# Reconstruction programs
cd recDir
make -f rec.mak
\rm *.o
# Evaluation programs
cd evalDir
make -f eval.mak
\rm *.o
# Training programs
cd trainDir
make -f train.mak
\rm *.o
# Utility programs
cd utilDir
make -f util.mak
\rm *.o
```

File: **recon.bat** MIPG / SF Jan94

```
# This batch:
# Reconstruct several phantoms using 4 methods
unalias cp
#/etc/renice 10 -u furuie
```

```
# Reconstruct
echo "Reconstruction log" > MIPGrecon.log
foreach i (010 011 012 013 014 015 )
echo "Processing case""$i"
echo "Processing ART blob"
artblob -b -i phaMIPG"$i".scn -f test -s 2 -n 1 -l 0.0055 -r 2 -m 2 -a 10.4
cat test.hist >> MIPGrecon.log
echo 'hostname' >> MIPGrecon.log
compImgs phaMIPG"$i".img test.img >> MIPGrecon.log
cp test.img MIPG"$i"ab.img
echo "Processing ART voxel"
artvox -b -i phaMIPG"$i".scn -f test -s 2 -n 1 -l 0.0049375
cat test.hist >> MIPGrecon.log
echo 'hostname' >> MIPGrecon.log
compImgs phaMIPG"$i".img test.img >> MIPGrecon.log
cp test.img MIPG"$i"av.img
echo "Processing EM blob"
emblob -b -i phaMIPG"$i".scn -f test -s 2 -n 7 -r 2 -m 2 -a 10.4 -t phaLengths.AttenF.scn
cat test.hist >> MIPGrecon.log
echo 'hostname' >> MIPGrecon.log
compImgs phaMIPG"$i".img test.img >> MIPGrecon.log
cp test.img MIPG"$i"eb.img
echo "Processing EM voxel"
em3d -b -i phaMIPG"$i".scn -f test -s 2 -n 3 -t phaLengths.AttenF.scn
cat test.hist >> MIPGrecon.log
echo 'hostname' >> MIPGrecon.log
compImgs phaMIPG"$i".img test.img >> MIPGrecon.log
cp test.img MIPG"$i"ev.img
end
alias cp 'cp -i'
```

File: **eval.bat**

```
#!/bin/csh
# Evaluation of several reconstructed images using 4 reconstruction methods
# File: eval.bat SF 20Dec93
# /etc/renice 10 -u furuie
# evaluate
foreach i ( 010 011 012 013 014 015 )
echo "Processing case""$i"
eval3d phaMIPG"$i".img MIPG"$i"ab.img testartblo.eval 4 2
eval3d phaMIPG"$i".img MIPG"$i"av.img testartvox.eval 4 2
eval3d phaMIPG"$i".img MIPG"$i"eb.img testemblo.eval 4 2
eval3d phaMIPG"$i".img MIPG"$i"ev.img testemvox.eval 4 2
end
```

File: **student.bat**

```
# file: student.bat SF
# Comparison of reconstruction methods based on t-test
\rm test.t
echo " ****ATTENTION: *****" > test.t
cat testartblo.eval | egrep "Detectability" >> test.t
# artblo vs. artvox
echo "*** artblo vs. artvox ***" >> test.t
echo " Table of t statistics: testartblo.eval.tbl VS testartvox.eval.tbl" >> test.t
echo " — HSD — — CSD — — SA —" >> test.t
echo "" >> test.t
echo -n "global " >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 3 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 4 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 2 0 >> test.t
```

```
echo "" >> test.t
echo -n "average " >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 14 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 15 0 >> test.t
echo "" >> test.t
echo -n "[4.0,6.4] " >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 8 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 11 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 5 0 >> test.t
echo "" >> test.t
echo -n "[8.0,10.]  " >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 9 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 12 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 6 0 >> test.t
echo "" >> test.t
echo -n "[16.,20.]  " >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 10 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 13 0 >> test.t
student testartblo.eval.tbl testartvox.eval.tbl 6 16 7 0 >> test.t
echo " " >> test.t
echo " " >> test.t
# artblo vs.  emblo
echo "*** artblo vs.  emblo ***" >> test.t
echo " Table of t statistics:  testartblo.eval.tbl VS testemblo.eval.tbl" >> test.t
echo " — HSD — — CSD — — SA —" >> test.t
echo "" >> test.t
echo -n "global " >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 3 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 4 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 2 0 >> test.t
echo "" >> test.t
echo -n "average " >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 14 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 15 0 >> test.t
echo "" >> test.t
echo -n "[4.0,6.4] " >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 8 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 11 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 5 0 >> test.t
echo "" >> test.t
echo -n "[8.0,10.]  " >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 9 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 12 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 6 0 >> test.t
echo "" >> test.t
echo -n "[16.,20.]  " >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 10 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 13 0 >> test.t
student testartblo.eval.tbl testemblo.eval.tbl 6 16 7 0 >> test.t
echo " " >> test.t
echo " " >> test.t
# artblo vs.  emvox
echo "*** artblo vs.  emvox ***" >> test.t
echo " Table of t statistics:  testartblo.eval.tbl VS testemvox.eval.tbl" >> test.t
echo " — HSD — — CSD — — SA —" >> test.t
echo "" >> test.t
echo -n "global " >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 3 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 4 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 2 0 >> test.t
```

```
echo "" >> test.t
echo -n "average " >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 14 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 15 0 >> test.t
echo "" >> test.t
echo -n "[4.0,6.4] " >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 8 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 11 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 5 0 >> test.t
echo "" >> test.t
echo -n "[8.0,10.]  " >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 9 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 12 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 6 0 >> test.t
echo "" >> test.t
echo -n "[16.,20.]  " >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 10 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 13 0 >> test.t
student testartblo.eval.tbl testemvox.eval.tbl 6 16 7 0 >> test.t
echo " " >> test.t
echo " " >> test.t
# artvox vs.  emblo
echo "*** artvox vs.  emblo ***" >> test.t
echo " Table of t statistics:  testartvox.eval.tbl VS testemblo.eval.tbl" >> test.t
echo " — HSD — — CSD — — SA —" >> test.t
echo "" >> test.t
echo -n "global " >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 3 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 4 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 2 0 >> test.t
echo "" >> test.t
echo -n "average " >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 14 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 15 0 >> test.t
echo "" >> test.t
echo -n "[4.0,6.4] " >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 8 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 11 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 5 0 >> test.t
echo "" >> test.t
echo -n "[8.0,10.]  " >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 9 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 12 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 6 0 >> test.t
echo "" >> test.t
echo -n "[16.,20.]  " >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 10 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 13 0 >> test.t
student testartvox.eval.tbl testemblo.eval.tbl 6 16 7 0 >> test.t
echo " " >> test.t
echo " " >> test.t
# artvox vs.  emvox
echo "*** artvox vs.  emvox ***" >> test.t
echo " Table of t statistics:  testartvox.eval.tbl VS testemvox.eval.tbl" >> test.t
echo " — HSD — — CSD — — SA —" >> test.t
echo "" >> test.t
echo -n "global " >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 3 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 4 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 2 0 >> test.t
```

```
echo "" >> test.t
echo -n "average " >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 14 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 15 0 >> test.t
echo "" >> test.t
echo -n "[4.0,6.4] " >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 8 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 11 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 5 0 >> test.t
echo "" >> test.t
echo -n "[8.0,10.] " >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 9 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 12 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 6 0 >> test.t
echo "" >> test.t
echo -n "[16.,20.] " >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 10 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 13 0 >> test.t
student testartvox.eval.tbl testemvox.eval.tbl 6 16 7 0 >> test.t
echo " " >> test.t
echo " " >> test.t
# emblo vs. emvox
echo "*** emblo vs. emvox ***" >> test.t
echo " Table of t statistics: testemblo.eval.tbl VS testemvox.eval.tbl" >> test.t
echo " — HSD — — CSD — — SA —" >> test.t
echo "" >> test.t
echo -n "global " >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 3 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 4 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 2 0 >> test.t
echo "" >> test.t
echo -n "average " >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 14 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 15 0 >> test.t
echo "" >> test.t
echo -n "[4.0,6.4] " >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 8 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 11 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 5 0 >> test.t
echo "" >> test.t
echo -n "[8.0,10.] " >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 9 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 12 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 6 0 >> test.t
echo "" >> test.t
echo -n "[16.,20.] " >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 10 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 13 0 >> test.t
student testemblo.eval.tbl testemvox.eval.tbl 6 16 7 0 >> test.t
echo " " >> test.t
echo " " >> test.t
# creating table
\rm test.tbt
echo " ****ATTENTION: using ROC area AVERAGE as FOM for detectability*****" > test.tbt
echo " Table output in file test.tbt"
echo " artblo artvox emblo emvox " >> test.tbt
#artblo
echo "" >> test.tbt
echo -n "artblo SA " >> test.tbt
echo -n " " >> test.tbt
```

```
student testartblo.eval.tbl testartvox.eval.tbl 6 16 2 0 >> test.tbt
student testartblo.eval.tbl testemblo.eval.tbl 6 16 2 0 >> test.tbt
student testartblo.eval.tbl testemvox.eval.tbl 6 16 2 0 >> test.tbt
echo "" >> test.tbt
echo -n " HSD" >> test.tbt
echo -n " " >> test.tbt
student testartblo.eval.tbl testartvox.eval.tbl 6 16 14 0 >> test.tbt
student testartblo.eval.tbl testemblo.eval.tbl 6 16 14 0 >> test.tbt
student testartblo.eval.tbl testemvox.eval.tbl 6 16 14 0 >> test.tbt
echo "" >> test.tbt
echo -n " CSD" >> test.tbt
echo -n " " >> test.tbt
student testartblo.eval.tbl testartvox.eval.tbl 6 16 15 0 >> test.tbt
student testartblo.eval.tbl testemblo.eval.tbl 6 16 15 0 >> test.tbt
student testartblo.eval.tbl testemvox.eval.tbl 6 16 15 0 >> test.tbt
echo "" >> test.tbt
#artvox
echo "" >> test.tbt
echo -n "artvox SA " >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
student testartvox.eval.tbl testemblo.eval.tbl 6 16 2 0 >> test.tbt
student testartvox.eval.tbl testemvox.eval.tbl 6 16 2 0 >> test.tbt
echo "" >> test.tbt
echo -n " HSD" >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
student testartvox.eval.tbl testemblo.eval.tbl 6 16 14 0 >> test.tbt
student testartvox.eval.tbl testemvox.eval.tbl 6 16 14 0 >> test.tbt
echo "" >> test.tbt
echo -n " CSD" >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
student testartvox.eval.tbl testemblo.eval.tbl 6 16 15 0 >> test.tbt
student testartvox.eval.tbl testemvox.eval.tbl 6 16 15 0 >> test.tbt
echo "" >> test.tbt
#emblo
echo "" >> test.tbt
echo -n "emblo SA " >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
student testemblo.eval.tbl testemvox.eval.tbl 6 16 2 0 >> test.tbt
echo "" >> test.tbt
echo -n " HSD" >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
student testemblo.eval.tbl testemvox.eval.tbl 6 16 14 0 >> test.tbt
echo "" >> test.tbt
echo -n " CSD" >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
echo -n " " >> test.tbt
student testemblo.eval.tbl testemvox.eval.tbl 6 16 15 0 >> test.tbt
echo "" >> test.tbt
```

The next two tables are examples of output of script **student.bat**.

Table 2 Table of t-statistic and level of significance associated with the comparison of pairs of five reconstruction techniques (example of output of script student.bat), using 12 phantoms (phaMIPG010...015, phaMIPG110...115) and equation (4) for detectability index and a summary table considering HSD and CSD as average of ROC area.

```
********** Detectability: (m1-m0).sqrt(n0*n1)/sqrt(n1*v1+n0*v0) *****
*** fbp vs. artblo ***
 Table of t statistics:  MIPG0fbp.eval.tbl  MIPG0artblo.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global      -0.297 [    ] -0.050 [    ] -4.041 [ --]
average      0.688 [    ]  0.307 [    ]
[4.0,6.4]    0.911 [    ]  0.176 [    ]  0.013 [    ]
[8.0,10.]    0.776 [    ]  1.490 [    ]  5.103 [+++]
[16.,20.]   -0.513 [    ] -0.943 [    ] -5.398 [---]

*** fbp vs. artvox ***
 Table of t statistics:  MIPG0fbp.eval.tbl  VS  MIPG0artvox.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global       3.881 [ ++]  2.690 [  +]  4.348 [ ++]
average      3.052 [  +]  4.805 [+++]
[4.0,6.4]    2.868 [  +]  2.725 [  +]  6.830 [+++]
[8.0,10.]    2.480 [  +]  3.720 [ ++] 12.806 [+++]
[16.,20.]   -0.653 [    ] -0.841 [    ] -1.938 [  -]

*** fbp vs. emblo ***
 Table of t statistics:  MIPG0fbp.eval.tbl  VS  MIPG0emblo.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global      -4.180 [ --] -1.625 [    ]  2.152 [  +]
average     -3.295 [ --] -2.172 [  -]
[4.0,6.4]   -1.012 [    ] -0.836 [    ]  6.285 [+++]
[8.0,10.]   -1.905 [  -]  0.034 [    ] 10.121 [+++]
[16.,20.]   -2.662 [  -] -3.621 [ --] -3.000 [  -]

*** fbp vs. emvox ***
 Table of t statistics:  MIPG0fbp.eval.tbl  VS  MIPG0emvox.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global       1.498 [    ]  0.416 [    ]  7.570 [+++]
average      0.731 [    ]  1.665 [    ]
[4.0,6.4]    1.554 [    ]  1.976 [  +]  8.331 [+++]
[8.0,10.]    1.655 [    ]  2.659 [  +] 14.863 [+++]
[16.,20.]   -2.725 [  -] -2.896 [  -]  0.570 [    ]

*** artblo vs. artvox ***
 Table of t statistics:  MIPG0artblo.eval.tbl  VS  MIPG0artvox.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global       5.674 [+++]  3.245 [ ++] 15.103 [+++]
average      4.076 [ ++]  4.587 [+++]
[4.0,6.4]    3.515 [ ++]  3.350 [ ++] 10.519 [+++]
[8.0,10.]    2.962 [  +]  4.268 [ ++] 13.626 [+++]
[16.,20.]   -0.123 [    ] -0.257 [    ]  9.771 [+++]

*** artblo vs. emblo ***
 Table of t statistics:  MIPG0artblo.eval.tbl  VS  MIPG0emblo.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global      -3.355 [ --] -2.151 [  -] 14.712 [+++]
average     -4.005 [ --] -3.097 [  -]
[4.0,6.4]   -2.406 [  -] -1.538 [    ] 12.294 [+++]
[8.0,10.]   -1.802 [  -] -1.287 [    ] 11.757 [+++]
[16.,20.]   -2.730 [  -] -2.991 [  -]  9.883 [+++]

*** artblo vs. emvox ***
 Table of t statistics:  MIPG0artblo.eval.tbl  VS  MIPG0emvox.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global       1.819 [  +]  0.492 [    ] 24.990 [+++]
average      0.284 [    ]  1.426 [    ]
[4.0,6.4]    1.250 [    ]  2.084 [  +] 13.466 [+++]
[8.0,10.]    1.063 [    ]  2.309 [  +] 17.612 [+++]
[16.,20.]   -2.735 [  -] -2.195 [  -] 15.545 [+++]

*** artvox vs. emblo ***
 Table of t statistics:  MIPG0artvox.eval.tbl  VS  MIPG0emblo.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global      -7.421 [---] -4.488 [---] -8.065 [---]
average     -7.401 [---] -8.331 [---]
[4.0,6.4]   -4.929 [---] -3.764 [ --] -5.356 [---]
[8.0,10.]   -3.198 [ --] -4.826 [---] -4.801 [---]
[16.,20.]   -3.707 [ --] -3.206 [ --] -2.515 [  -]

*** artvox vs. emvox ***
 Table of t statistics:  MIPG0artvox.eval.tbl  VS  MIPG0emvox.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global      -3.840 [ --] -4.614 [---]  9.184 [+++]
average     -4.805 [---] -4.237 [ --]
[4.0,6.4]   -2.963 [  -] -0.860 [    ]  1.652 [    ]
[8.0,10.]   -2.415 [  -] -2.551 [  -]  3.637 [ ++]
[16.,20.]   -3.903 [ --] -2.765 [  -]  7.062 [+++]

*** emblo vs. emvox ***
 Table of t statistics:  MIPG0emblo.eval.tbl  VS  MIPG0emvox.eval.tbl
          --- HSD ---     --- CSD ---   --- SA ---

global       6.912 [+++]  2.637 [  +] 24.389 [+++]
average      3.836 [ ++]  5.235 [+++]
[4.0,6.4]    3.600 [ ++]  3.415 [ ++] 14.122 [+++]
[8.0,10.]    2.620 [  +]  3.296 [ ++] 26.047 [+++]
[16.,20.]    0.719 [    ]  2.155 [  +] 11.056 [+++]

 ****ATTENTION:  using ROC area AVERAGE as FOM for detectability*****
                 artblo          artvox          emblo           emvox
fbp     SA  -4.041 [ --]  4.348 [ ++]  2.152 [  +]  7.570 [+++]
        HSD  0.688 [    ]  3.052 [  +] -3.295 [ --]  0.731 [    ]
        CSD  0.307 [    ]  4.805 [+++] -2.172 [  -]  1.665 [    ]

artblo  SA                 15.103 [+++] 14.712 [+++] 24.990 [+++]
        HSD                 4.076 [ ++] -4.005 [ --]  0.284 [    ]
        CSD                 4.587 [+++] -3.097 [  -]  1.426 [    ]

artvox  SA                              -8.065 [---]  9.184 [+++]
        HSD                             -7.401 [---] -4.805 [---]
        CSD                             -8.331 [---] -4.237 [ --]

emblo   SA                                           24.389 [+++]
        HSD                                           3.836 [ ++]
        CSD                                           5.235 [+++]
```

Table 3 Table of t-statistic and level of significance associated with the comparison of pairs of five reconstruction techniques (example of output of script student.bat), using 12 phantoms (phaMIPG010...015, phaMIPG110...115) and equation (5) for detectability index and a summary table considering HSD and CSD as average of ROC area.

```
****************** Detectability: (m1-m0) ******************
*** fbp vs. artblo ***
 Table of t statistics:  MIPGfbp.eval.tbl  MIPGartblo.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global      1.638 [    ]  3.224 [ ++] -4.041 [ --]
average     0.725 [    ]  1.236 [    ]
[4.0,6.4]   1.489 [    ]  1.732 [    ]  0.013 [    ]
[8.0,10.]  -0.250 [    ]  0.661 [    ]  5.103 [+++]
[16.,20.]  -0.620 [    ] -1.142 [    ] -5.398 [---]

*** fbp vs. artvox ***
 Table of t statistics:  MIPGfbp.eval.tbl  VS  MIPGartvox.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global      4.345 [ ++]  5.772 [+++]  4.348 [ ++]
average     3.038 [  +]  4.601 [+++]
[4.0,6.4]   3.313 [ ++]  3.905 [ ++]  6.830 [+++]
[8.0,10.]   1.996 [  +]  3.644 [ ++] 12.806 [+++]
[16.,20.]  -0.784 [    ] -0.826 [    ] -1.938 [  -]

*** fbp vs. emblo ***
 Table of t statistics:  MIPGfbp.eval.tbl  VS  MIPGemblo.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global     -1.618 [    ] -0.059 [    ]  2.152 [  +]
average    -2.407 [  -] -2.798 [  -]
[4.0,6.4]   0.506 [    ] -0.577 [    ]  6.285 [+++]
[8.0,10.]  -2.027 [  -] -0.396 [    ] 10.121 [+++]
[16.,20.]  -3.237 [ --] -4.341 [ --] -3.000 [  -]

*** fbp vs. emvox ***
 Table of t statistics:  MIPGfbp.eval.tbl  VS  MIPGemvox.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global      2.221 [  +]  3.598 [ ++]  7.570 [+++]
average     0.989 [    ]  2.344 [  +]
[4.0,6.4]   2.936 [  +]  3.138 [ ++]  8.331 [+++]
[8.0,10.]   1.093 [    ]  2.474 [  +] 14.863 [+++]
[16.,20.]  -3.303 [ --] -3.033 [  -]  0.570 [    ]

*** artblo vs. artvox ***
 Table of t statistics:  MIPGartblo.eval.tbl  VS  MIPGartvox.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global      4.406 [ ++]  5.303 [+++] 15.103 [+++]
average     3.282 [ ++]  4.903 [+++]
[4.0,6.4]   3.761 [ ++]  4.088 [ ++] 10.519 [+++]
[8.0,10.]   2.425 [  +]  4.301 [ ++] 13.626 [+++]
[16.,20.]  -0.298 [    ]  0.121 [    ]  9.771 [+++]

*** artblo vs. emblo ***
 Table of t statistics:  MIPGartblo.eval.tbl  VS  MIPGemblo.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global     -5.433 [---] -3.363 [ --] 14.712 [+++]
average    -4.755 [---] -3.659 [ --]
[4.0,6.4]  -1.582 [    ] -2.399 [  -] 12.294 [+++]
[8.0,10.]  -2.734 [  -] -0.869 [    ] 11.757 [+++]
[16.,20.]  -3.093 [  -] -3.024 [  -]  9.883 [+++]

*** artblo vs. emvox ***
 Table of t statistics:  MIPGartblo.eval.tbl  VS  MIPGemvox.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global      1.763 [    ]  2.630 [  +] 24.990 [+++]
average     0.724 [    ]  1.892 [  +]
[4.0,6.4]   2.124 [  +]  2.430 [  +] 13.466 [+++]
[8.0,10.]   1.472 [    ]  2.740 [  +] 17.612 [+++]
[16.,20.]  -3.472 [ --] -2.462 [  -] 15.545 [+++]

*** artvox vs. emblo ***
 Table of t statistics:  MIPGartvox.eval.tbl  VS  MIPGemblo.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global     -9.724 [---] -7.971 [---] -8.065 [---]
average    -8.125 [---] -8.246 [---]
[4.0,6.4]  -4.429 [ --] -5.228 [---] -5.356 [---]
[8.0,10.]  -3.859 [ --] -5.135 [---] -4.801 [---]
[16.,20.]  -4.746 [---] -3.063 [  -] -2.515 [  -]

*** artvox vs. emvox ***
 Table of t statistics:  MIPGartvox.eval.tbl  VS  MIPGemvox.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global     -4.376 [ --] -2.886 [  -]  9.184 [+++]
average    -4.602 [---] -3.070 [  -]
[4.0,6.4]  -1.136 [    ] -0.721 [    ]  1.652 [    ]
[8.0,10.]  -2.204 [  -] -2.448 [  -]  3.637 [ ++]
[16.,20.]  -4.681 [---] -2.787 [  -]  7.062 [+++]

*** emblo vs. emvox ***
 Table of t statistics:  MIPGemblo.eval.tbl  VS  MIPGemvox.eval.tbl
          --- HSD ---     --- CSD ---  --- SA ---

global      5.868 [+++]  5.121 [+++] 24.389 [+++]
average     4.889 [+++]  5.564 [+++]
[4.0,6.4]   4.131 [ ++]  4.028 [ ++] 14.122 [+++]
[8.0,10.]   3.722 [ ++]  3.193 [ ++] 26.047 [+++]
[16.,20.]   0.415 [    ]  0.729 [    ] 11.056 [+++]

 ****ATTENTION:  using ROC area AVERAGE as FOM for detectability*****
              artblo        artvox        emblo         emvox
fbp     SA  -4.041 [ --]  4.348 [ ++]  2.152 [  +]  7.570 [+++]
        HSD  0.725 [    ]  3.038 [  +] -2.407 [  -]  0.989 [    ]
        CSD  1.236 [    ]  4.601 [+++] -2.798 [  -]  2.344 [  +]

artblo  SA               15.103 [+++] 14.712 [+++] 24.990 [+++]
        HSD               3.282 [ ++] -4.755 [---]  0.724 [    ]
        CSD               4.903 [+++] -3.659 [ --]  1.892 [  +]

artvox  SA                            -8.065 [---]  9.184 [+++]
        HSD                           -8.125 [---] -4.602 [---]
        CSD                           -8.246 [---] -3.070 [  -]

emblo   SA                                        24.389 [+++]
        HSD                                        4.889 [+++]
        CSD                                        5.564 [+++]
```

Appendix G FOMs similarity

Program **eval3d** gives two option for detectability index formula (equation (4) and equation (5) [see section 3.3]). For each formula, it provides ROC area (FOM, detectability) values per feature size (small, medium, large), average of these three ROC areas (_avg), and a global ROC area (_gl) (see 2.4.3). However, there are some similarities among these FOMs. Figures 10 and 11 show the rank-ordering similarity (Yeung and Herman, 1989) for hot spot detectability and cold spot detectability, respectively. For a defined FOM, after a statistical comparison (student test) between two reconstruction techniques over a set of phantoms (12 phantoms: phaMIPG010..015, phaMIPG110...115), we have a statistic (t). In our case, we had 10 comparisons (fbp, artblob, artvox, emblob, emvox). Now computing the rank-ordering similarity between two specified FOMs using t-values, we have an entry in the tables.

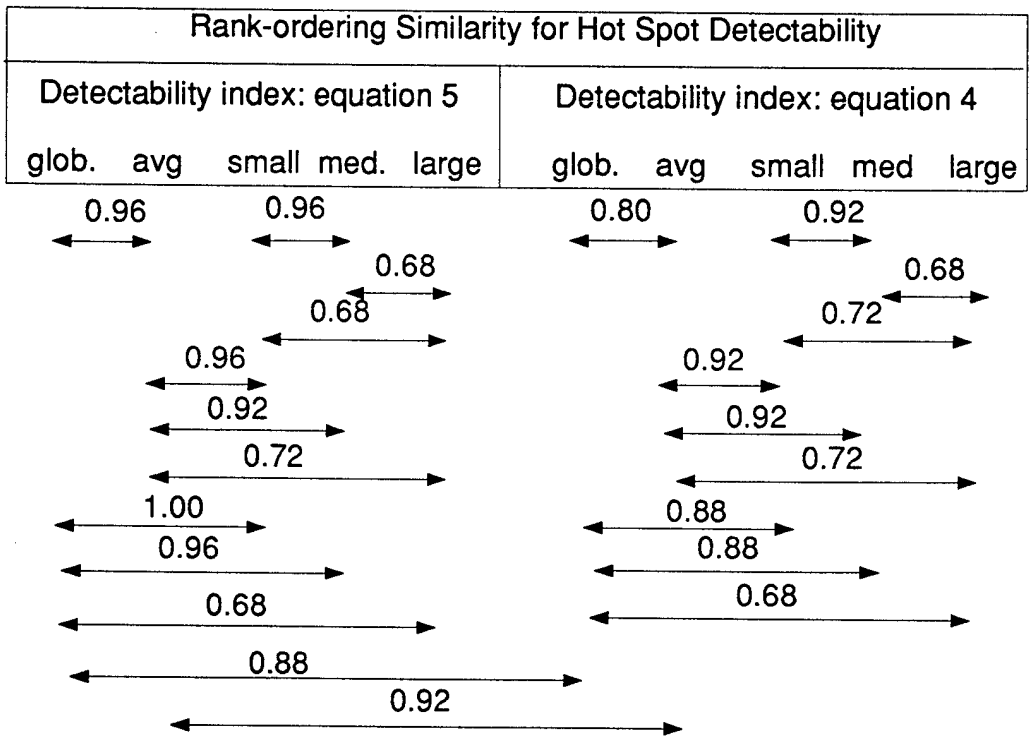


Figure 10. Rank-ordering similarity for Hot Spot Detectability

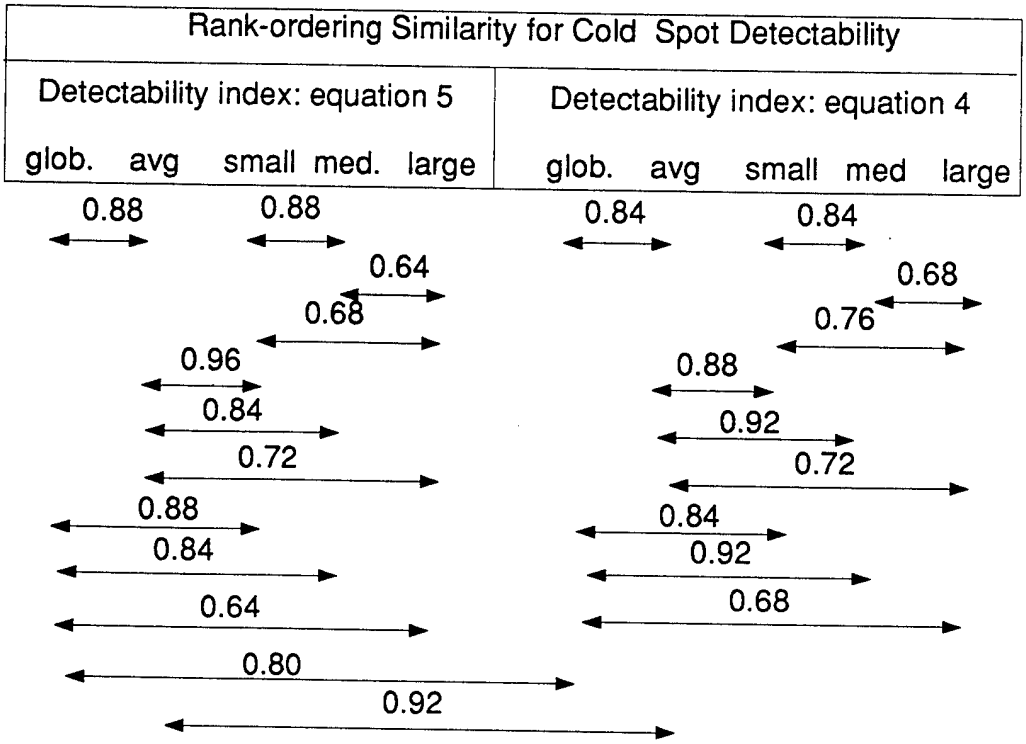


Figure 11. Rank-ordering similarity for Cold Spot Detectability

The expected value and standard deviation of similarity variable is (for N=10) 0.34 and 0.1426 respectively. Therefore, we can conclude that there are statistically significant similarity between several FOMs, in special between average of ROC areas (avg) using equation 5 and 4; average of ROC areas and ROC area of small features; and global ROC area and ROC area of small features. The tables also show that small features have stronger influence on the calculation of global and average ROC areas than medium and large sizes for the studied phantoms.